ITSE 1345 PL/SQL Lab 4

Elizabeth Arlene Waghalter

0ae1c910-fa07-4326-978c-2e1484b06b58

1. Create a procedure which will determine whether a given range of customers have purchase orders. Use explicit cursors to process the table data.

Specifications:

Inputs: 1. low Cust\_ID value for Cust\_ID range.

2. high Cust\_ID value for Cust\_ID range.

Pre: first Cust\_ID parameter < second Cust\_ID parameter; if not, use appropriate error message.

Post: Order by Cust\_ID.

Print Cust\_ID with appropriate message: ‘Has a purchase order’ or ‘does not have a purchase order.’

Write a driver to call the procedure.

Test these ranges: 90001-90008

90003-90007

90009-90010

90005-90004

select \* from CUSTOMER\_LAB1;

CUST\_ID CUST\_NAME ACCOUNT\_ID AC ST

---------- --------------- ---------- -- --

90001 B and B A-11101 PR AK

90002 XYZ A-11102 CM NJ

90003 JJ Sons A-11103 CM NJ

90004 Exxon A-11104 PR NY

90005 ABC A-11105 CM NY

90006 Smith Co. A-11106 CM MD

90007 Brown Co. A-11107 CM MD

90008 Cooper Inc. A-11108 PR MD

8 rows selected

select \* from purchase\_order;

ORDER\_ID WIDGET\_ID CUST\_ID QUANTITY PURCHASE\_DATE PURCHASE\_PRICE D

---------- ---------- ---------- ---------- ------------- -------------- -

31 61 90001 10 21-JAN-00 100 N

32 63 90004 10 11-FEB-98 100 N

33 64 90008 8 20-MAR-99 80 N

34 65 90007 7 10-JUN-99 90 N

35 62 90004 12 25-JUN-98 75 Y

36 65 90001 9 05-JAN-00 90 Y

37 61 90005 8 15-JUL-99 100 N

38 63 90001 5 22-JUL-98 100 Y

39 62 90005 7 01-AUG-98 75 N

40 64 90008 10 20-AUG-99 80 Y

10 rows selected

CREATE OR REPLACE PROCEDURE cust\_po

(pv\_low IN NUMBER,

pv\_high IN NUMBER)

IS

CURSOR cust\_cur(cv\_low NUMBER, cv\_high NUMBER)IS

SELECT cust\_id, purchase\_date

FROM PURCHASE\_ORDER

WHERE cust\_id BETWEEN cv\_low AND cv\_high

ORDER BY cust\_id;

cust\_var NUMBER;

d\_var DATE;

BEGIN

OPEN cust\_cur(pv\_low, pv\_high);

LOOP

FETCH cust\_cur INTO cust\_var, d\_var;

EXIT WHEN cust\_cur%NOTFOUND;

IF pv\_low > pv\_high THEN

dbms\_output.put\_line('First var must be smaller than second var.');

ELSE

IF d\_var IS NOT NULL THEN

dbms\_output.put\_line('Customer ' ||cust\_var|| ' has a purchase order.');

ELSE

dbms\_output.put\_line('Customer ' ||cust\_var|| ' does not have a purchase order.');

END IF;

END IF;

END LOOP;

CLOSE cust\_cur;

END;

/

Procedure CUST\_PO compiled

BEGIN

cust\_po(90001, 90008);

END;

/

PL/SQL procedure successfully completed.

Customer 90001 has a purchase order.

Customer 90001 has a purchase order.

Customer 90001 has a purchase order.

Customer 90004 has a purchase order.

Customer 90004 has a purchase order.

Customer 90005 has a purchase order.

Customer 90005 has a purchase order.

Customer 90007 has a purchase order.

Customer 90008 has a purchase order.

Customer 90008 has a purchase order.

BEGIN

cust\_po(90003, 90007);

END;

/

PL/SQL procedure successfully completed.

Customer 90004 has a purchase order.

Customer 90004 has a purchase order.

Customer 90005 has a purchase order.

Customer 90005 has a purchase order.

Customer 90007 has a purchase order.

BEGIN

cust\_po(90009, 90010);

END;

/

PL/SQL procedure successfully completed.

BEGIN

cust\_po(90005, 90004);

END;

/

PL/SQL procedure successfully completed.

2. Create a package named HOSPITAL which operates on the PHYSICIAN, TREATMENT, AND PATIENT tables. Create the tables first.

The package should contain:

-a PL/SQL RECORD table named t\_patTrt defined in the specification with the following columns – Pat\_Nbr, Trt\_Procedure, Phys\_ID, Phys\_Name, Phys\_Specialty.

-an exception named e\_DupPhysFound.

-a procedure named BuildPatTbl that will build the treatment table for all patients:

TABLE OUT

ROWCOUNT IN OUT

INDEX starts at 1, incremented by one

-2 overloaded functions, both named FindPatient that can check the db by patient number or name. RETURN true if patient found; false if not found.

-a procedure named NewPhys which inserts a new physician into the physician table. Input parameters(Phys\_ID, Phys\_Name, Phys\_Phone, Phys\_Specialty). Check if physician is already in the table. If physician is in the table, raise the exception e\_DupPhysFound.

CREATE OR REPLACE PACKAGE hospital IS

PRAGMA SERIALLY\_REUSABLE;

TYPE t\_patTrt IS RECORD

(Pat\_Nbr NUMBER,

Trt\_Procedure NUMBER,

Phys\_ID physician.phys\_ID%TYPE,

Phys\_Name physician.phys\_name%TYPE,

Phys\_Specialty physician.phys\_specialty%TYPE);

e\_DupPhysFound EXCEPTION;

PRAGMA EXCEPTION\_INIT(e\_DupPhysFound, -20001);

FUNCTION findPatient

(Pat\_Nbr IN NUMBER)

RETURN VARCHAR2;

FUNCTION findPatient

(Pat\_Name IN VARCHAR2)

RETURN VARCHAR2;

PROCEDURE BuildPatTbl

(r\_coun OUT NUMBER,

ind\_var IN NUMBER);

PROCEDURE NewPhys

(Phys\_ID IN physician.phys\_ID%TYPE);

END HOSPITAL;

/

Package HOSPITAL compiled

CREATE OR REPLACE PACKAGE BODY hospital

IS

PRAGMA SERIALLY\_REUSABLE;

FUNCTION findPatient(p\_num NUMBER)

RETURN VARCHAR2

IS

CURSOR pat\_cur IS

SELECT \* FROM PATIENT

WHERE pat\_nbr = p\_num;

ret\_val PAT\_CUR%ROWTYPE;

BEGIN

OPEN pat\_cur;

LOOP

FETCH pat\_cur INTO ret\_val;

EXIT WHEN pat\_cur%NOTFOUND;

IF ret\_val.pat\_nbr = p\_num THEN

dbms\_output.put\_line(ret\_val.pat\_nbr);

dbms\_output.put\_line(ret\_val.pat\_name);

dbms\_output.put\_line(ret\_val.pat\_address);

dbms\_output.put\_line(ret\_val.pat\_city);

dbms\_output.put\_line(ret\_val.pat\_state);

dbms\_outuput.put\_line(ret\_val.pat\_zip);

dbms\_output.put\_line(ret\_val.pat\_room);

dbms\_output.put\_line(ret\_val.pat\_bed);

ELSE

dbms\_output.put\_line('Customer ID not valid.');

END IF;

END LOOP;

CLOSE pat\_cur;

END findPatient;

FUNCTION findPatient(p\_name VARCHAR2)

RETURN VARCHAR2

IS

CURSOR pat\_cur IS

SELECT \* FROM PATIENT

WHERE pat\_name = p\_name;

ret\_val PAT\_CUR%ROWTYPE;

BEGIN

OPEN pat\_cur;

LOOP

FETCH pat\_cur INTO ret\_val;

EXIT WHEN pat\_cur%NOTFOUND;

IF ret\_val.pat\_name = p\_name THEN

dbms\_output.put\_line(ret\_val.pat\_nbr);

dbms\_output.put\_line(ret\_val.pat\_name);

dbms\_output.put\_line(ret\_val.pat\_address);

dbms\_output.put\_line(ret\_val.pat\_city);

dbms\_output.put\_line(ret\_val.pat\_state);

dbms\_outuput.put\_line(ret\_val.pat\_zip);

dbms\_output.put\_line(ret\_val.pat\_room);

dbms\_output.put\_line(ret\_val.pat\_bed);

ELSE

dbms\_output.put\_line('Customer name not valid.');

END IF;

END LOOP;

CLOSE pat\_cur;

END findPatient;

PROCEDURE buildpattbl IS

p\_id VARCHAR2(1000);

BEGIN

p\_id := 'CREATE TABLE PATIENT

(Pat\_Nbr NUMBER,

Pat\_Name VARCHAR2,

Pat\_Address VARCHAR2,

Pat\_City VARCHAR2,

Pat\_State VARCHAR2,

Pat\_Zip VARCHAR2,

Pat\_Room NUMBER,

Pat\_Bed NUMBER)';

EXECUTE IMMEDIATE p\_id;

BEGIN

INSERT INTO Patient

VALUES(1379, 'Cribbs, John', '2110 Main St.', 'Austin', 'TX', '78711', 101, 1);

INSERT INTO Patient

VALUES(3249, 'Baker, Mary', '3547 W. 42nd St.', 'Berkeley', 'CA', '94117', 137, 2);

INSERT INTO Patient

VALUES(4500, 'Garcia, Juan', '1533 Telegraph', 'Berkeley', 'CA', '94117', 228, 2);

INSERT INTO Patient

VALUES(5116, 'Harris, Carol', '4710 Ave. E', 'Austin', 'TX', '78705', 438, 1);

INSERT INTO Patient

VALUES(5872, 'Zimmer, Elka', '7988 Cedar', 'Cleveland', 'OH', '44060', 137, 1);

INSERT INTO Patient

VALUES(6213, 'Rose, David', '322 Bridge Ave.', 'Redwood', 'CA', '94065', 100, 1);

INSERT INTO Patient

VALUES(7459, 'Smith, Chris', '788 Cummings', 'Cleveland', 'OH', '44066', 438, 3);

INSERT INTO Patient

VALUES(8031, 'Fitch, Sylvia', '3380 Fox Ave.', 'Madison', 'WI', '53711', 420, 4);

INSERT INTO Patient

VALUES(8659, 'Hernandez, Juan', '8300 Geneva Dr.', 'Austin', 'TX', '78723', 350, 2);

END buildpattbl;

PROCEDURE NewPhys is

(p\_phys\_id NUMBER,

p\_phys\_name VARCHAR2,

p\_phys\_phone VARCHAR2,

p\_phys\_specialty VARCHAR2)

BEGIN

INSERT INTO Physician

VALUES(101, 'Wilcox, Chris', '512-329-1848', 'Ears, Nose, Throat');

INSERT INTO Physician

VALUES(102, 'Nusca, Jane', '512-516-3947', 'Cardiovascular');

INSERT INTO Physician

VALUES(103, 'Gomez, Juan', '512-382-4987', 'Orthopedics');

INSERT INTO Physician

VALUES(104, 'Li, Jan', '512-516-3984', 'Cardiovascular');

INSERT INTO Physician

VALUES(105, 'Simmons, Alex', '512-442-5700', 'Hematology');

END NewPhys;

END hospital;

/

Package body HOSPITAL compiled

Errors: check compiler log

select \* from physician;

PHYS\_ID PHYS\_NAME PHYS\_PHONE PHYS\_SPECIALTY

---------- -------------------------------------------------- --------------- -------------------------

102 Nusca, Jane 512-516-3947 Cardiovascular

103 Gomez, Juan 512-382-4987 Orthopedics

104 Li, Jan 512-516-3984 Cardiovascular

105 Simmons, Alex 512-442-5700 Hematology

4. Modify and add data to the Physician, Patient, and Treatment tables to resemble those provided.

select \* from physician

order by PHYS\_ID;

PHYS\_ID PHYS\_NAME PHYS\_PHONE PHYS\_SPECIALTY

---------- -------------------------------------------------- --------------- -------------------------

101 Wilcox, Chris 512-329-1848 Eyes, Ears, Throat

102 Nusca, Jane 512-516-3947 Cardiovascular

103 Gomez, Juan 512-382-4987 Orthopedics

104 Li, Jan 512-516-3984 Cardiovascular

105 Simmons, Alex 512-442-5700 Hematology

CREATE TABLE Patient

(Pat\_Nbr NUMBER(5),

Pat\_Name VARCHAR2(35),

Pat\_Address VARCHAR2(45),

Pat\_City VARCHAR2(35),

Pat\_State VARCHAR2(2),

Pat\_ZIP VARCHAR2(10),

Pat\_Room NUMBER(3),

Pat\_Bed NUMBER(1));

Table PATIENT created.

INSERT INTO Patient

VALUES(1379, 'Cribbs, John', '2110 Main St.', 'Austin', 'TX', '78711', 101, 1);

INSERT INTO Patient

VALUES(3249, 'Baker, Mary', '3547 W. 42nd St.', 'Berkeley', 'CA', '94117', 137, 2);

INSERT INTO Patient

VALUES(4500, 'Garcia, Juan', '1533 Telegraph', 'Berkeley', 'CA', '94117', 228, 2);

INSERT INTO Patient

VALUES(5116, 'Harris, Carol', '4710 Ave. E', 'Austin', 'TX', '78705', 438, 1);

INSERT INTO Patient

VALUES(5872, 'Zimmer, Elka', '7988 Cedar', 'Cleveland', 'OH', '44060', 137, 1);

INSERT INTO Patient

VALUES(6213, 'Rose, David', '322 Bridge Ave.', 'Redwood', 'CA', '94065', 100, 1);

INSERT INTO Patient

VALUES(7459, 'Smith, Chris', '788 Cummings', 'Cleveland', 'OH', '44066', 438, 3);

INSERT INTO Patient

VALUES(8031, 'Fitch, Sylvia', '3380 Fox Ave.', 'Madison', 'WI', '53711', 420, 4);

INSERT INTO Patient

VALUES(8659, 'Hernandez, Juan', '8300 Geneva Dr.', 'Austin', 'TX', '78723', 350, 2);

select \* from patient;

PAT\_NBR PAT\_NAME PAT\_ADDRESS PAT\_CITY PA PAT\_ZIP PAT\_ROOM PAT\_BED

---------- ----------------------------------- --------------------------------------------- ----------------------------------- --

1379 Cribbs, John 2110 Main St. Austin TX 78711 101 1

3249 Baker, Mary 3547 W. 42nd St. Berkeley CA 94117 137 2

4500 Garcia, Juan 1533 Telegraph Berkeley CA 94117 228 2

5116 Harris, Carol 4710 Ave. E Austin TX 78705 438 1

5872 Zimmer, Elka 7988 Cedar Cleveland OH 44060 137 1

6213 Rose, David 322 Bridge Ave. Redwood CA 94065 100 1

7459 Smith, Chris 788 Cummings Cleveland OH 44066 438 3

8031 Fitch, Sylvia 3380 Fox Ave. Madison WI 53711 420 4

8659 Hernandez, Juan 8300 Geneva Dr. Austin TX 78723 P350 2

9 rows selected

CREATE TABLE t\_patTrt

(Pat\_Nbr NUMBER(4),

Phys\_ID NUMBER(3),

Trt\_Procedure NUMBER(5),

Trt\_Date DATE);

Table T\_PATTRT created.

INSERT INTO T\_PATTRT

VALUES(3249,101,'13-08', '12-FEB-1999');

INSERT INTO T\_PATTRT

VALUES(1379,103,'27-45', '25-MAR-1999');

INSERT INTO T\_PATTRT

VALUES(3249,103, '88-20', '22-JAN-1999');

INSERT INTO T\_PATTRT

VALUES(5116, 104, '52-14', '03-APR-1999');

INSERT INTO T\_PATTRT

VALUES(4500, 101, '13-08', '04-FEB-1999');

INSERT INTO T\_PATTRT

VALUES(8031, 102, '52-14', '15-MAR-2000');

INSERT INTO T\_PATTRT

VALUES(5116, 104, '52-14', '05-FEB-2001');

INSERT INTO T\_PATTRT

VALUES(5872, 105, '60-00', '13-FEB-2000');

INSERT INTO T\_PATTRT

VALUES(3249, 103, '88-20', '24-JAN-2000');

INSERT INTO T\_PATTRT

VALUES(8659, 104, '60-00', '08-APR-2001');

SELECT \* FROM T\_PATTRT;

PAT\_NBR PHYS\_ID TRT\_P TRT\_DATE

---------- ---------- ----- ---------

3249 101 13-08 12-FEB-99

1379 103 27-45 25-MAR-99

3249 103 88-20 22-JAN-99

5116 104 52-14 03-APR-99

4500 101 13-08 04-FEB-99

8031 102 52-14 15-MAR-00

5116 104 52-14 05-FEB-01

5872 105 60-00 13-FEB-00

3249 103 88-20 24-JAN-00

8659 104 60-00 08-APR-01

10 rows selected

CREATE TABLE procedure

(Pro\_Nbr VARCHAR2(5),

Pro\_Desc VARCHAR2(25),

Pro\_Charge DECIMAL(6,2));

Table PROCEDURE created.

INSERT INTO PROCEDURE

VALUES('13-08', 'Throat culture',15.00);

INSERT INTO PROCEDURE

VALUES('27-45', 'X-Ray',62.00);

INSERT INTO PROCEDURE

VALUES('52-14', 'Cardiogram',135.00);

INSERT INTO PROCEDURE

VALUES('60-00', 'Blood Analysis', 58.00);

INSERT INTO PROCEDURE

VALUES('88-20', 'MRI', 800.00);

SELECT \* FROM PROCEDURE;

PRO\_N PRO\_DESC PRO\_CHARGE

----- ------------------------- ----------

13-08 Throat culture 15

27-45 X-Ray 62

52-14 Cardiogram 135

60-00 Blood Analysis 58

88-20 MRI 800

5. Create a trigger for inserting into or updating the treatment table, which checks the treatment date to ensure the date range is between 3 months ago to today. Raise a application error (-20001, ‘Invalid Treatment date’).

CREATE OR REPLACE TRIGGER ch\_dat

BEFORE INSERT OR UPDATE

ON T\_PATTRT

BEGIN

DECLARE

V\_DATE DATE;

S\_DATE DATE;

BEGIN

SELECT SYSDATE, ADD\_MONTHS(SYSDATE, -3) INTO V\_DATE, S\_DATE FROM DUAL;

END;

END;

/

Trigger CH\_DAT compiled

6. a. Create a new table called Trt\_Stats to keep track of the number of times each Trt\_Procedure is modified (INSERT, DELETE, UPDATE):

CREATE TABLE Trt\_Stats

(Trt\_Procedure VARCHAR2(35),

Trt\_INS\_Count NUMBER(5),

Trt\_DEL\_Count NUMBER(5),

Trt\_UPD\_Count NUMBER(5));

Table TRT\_STATS created.

DESCRIBE TRT\_STATS;

Name Null Type

------------- ---- ------------

TRT\_PROCEDURE VARCHAR2(35)

TRT\_INS\_COUNT NUMBER(5)

TRT\_DEL\_COUNT NUMBER(5)

TRT\_UPD\_COUNT NUMBER(5)

b. Write a trigger for the Treatment Table that does the following:

~for each INSERT, add 1 to the Trt\_INS\_Count

~for each DELETE, add 1 to the Trt\_DEL\_Count

~for each UPDATE, add 1 to the Trt\_UPD\_Count

~test

create or replace TRIGGER change\_count

AFTER INSERT OR DELETE OR UPDATE ON Trt\_Stats

DECLARE

ins\_var number:=0;

del\_var number:=0;

up\_var number :=0;

BEGIN

IF INSERTING THEN

ins\_var := ins\_var +1;

dbms\_output.put\_line(ins\_var);

ELSIF DELETING THEN

del\_var := del\_var +1;

dbms\_output.put\_line(del\_var);

ELSIF UPDATING THEN

up\_var := up\_var +1;

dbms\_output.put\_line(up\_var);

END IF;

END;

Trigger CHANGE\_COUNT compiled

INSERT INTO TRT\_STATS(TRT\_PROCEDURE)

VALUES('Aspiration');

1 row inserted.

INSERT INTO TRT\_STATS(TRT\_PROCEDURE)

VALUES('Surgery Prep');

1 row inserted.

INSERT INTO TRT\_STATS(TRT\_PROCEDURE)

VALUES('Mouth swab');

1 row inserted.

TRT\_PROCEDURE TRT\_INS\_COUNT TRT\_DEL\_COUNT TRT\_UPD\_COUNT

----------------------------------- ------------- ------------- -------------

Aspiration

Surgery Prep

Mouth swab