|  |
| --- |
| ADDB7311 |
| Assignment 2 |
| Robert Andrew Hurst |

|  |
| --- |
| 13005890  5-24-2017 |

DROP TABLE TREATMENT\_TABLE;

DROP TABLE BILLING\_TABLE;

DROP TABLE MEDICINE\_TABLE;

DROP TABLE DISEASE\_TABLE;

DROP TABLE STAFF\_TABLE;

DROP TABLE VETERINARIAN\_TABLE;

DROP TABLE SPECIALITY\_TABLE;

DROP TABLE ANIMAL\_TABLE;

DROP TABLE CUSTOMER\_TABLE;

CREATE TABLE CUSTOMER\_TABLE (

CUSTOMER\_ID VARCHAR(6),

FIRST\_NAME VARCHAR(50),

SURNAME VARCHAR(50),

CONTACT VARCHAR(30),

EMAIL VARCHAR(30),

CONSTRAINT PK\_CUSTOMER\_ID PRIMARY KEY (CUSTOMER\_ID)

);

CREATE TABLE ANIMAL\_TABLE (

ANIMAL\_ID VARCHAR(4),

ANIMAL\_TYPE VARCHAR(30),

ANIMAL\_BREED VARCHAR(30),

ANIMAL\_COLOUR VARCHAR(30),

CONSTRAINT PK\_ANIMAL\_ID PRIMARY KEY (ANIMAL\_ID)

);

CREATE TABLE SPECIALITY\_TABLE (

SPECIALTY\_ID INT,

SPECIALITY VARCHAR(30),

CONSTRAINT PK\_SPECAILITY\_ID PRIMARY KEY (SPECIALTY\_ID)

);

CREATE TABLE VETERINARIAN\_TABLE (

VET\_ID VARCHAR(4),

FIRST\_NAME VARCHAR(50),

SURNAME VARCHAR(50),

CONTACT VARCHAR(30),

EMAIL VARCHAR(30),

SPECIALITY\_ID INT,

CONSTRAINT PK\_VET\_ID PRIMARY KEY (VET\_ID),

CONSTRAINT FK\_SPECIALITY\_ID FOREIGN KEY (SPECIALITY\_ID) REFERENCES SPECIALITY\_TABLE (SPECIALTY\_ID)

);

CREATE TABLE STAFF\_TABLE (

STAFF\_ID VARCHAR(4),

FIRST\_NAME VARCHAR(50),

SURNAME VARCHAR(50),

JOB\_DESCRIPTION VARCHAR(50),

CONTACT VARCHAR(30),

CONSTRAINT PK\_STAFF\_ID PRIMARY KEY (STAFF\_ID)

);

CREATE TABLE DISEASE\_TABLE (

DISEASE\_ID VARCHAR(9),

DISEASE\_NAME VARCHAR(50),

CONSTRAINT PK\_DISEASE\_ID PRIMARY KEY (DISEASE\_ID)

);

CREATE TABLE MEDICINE\_TABLE (

MED\_ID VARCHAR(9),

MED\_NAME VARCHAR(50),

MED\_PRICE NUMBER,

MED\_EXPIRY\_DATE DATE,

CONSTRAINT PK\_MED\_ID PRIMARY KEY (MED\_ID)

);

CREATE TABLE BILLING\_TABLE (

BILL\_ID VARCHAR(9),

CUSTOMER\_ID VARCHAR(6),

STAFF\_ID VARCHAR(4),

BILL\_HRS NUMBER,

CONSTRAINT PK\_BILL\_ID PRIMARY KEY (BILL\_ID),

CONSTRAINT FK\_CUSTOMER\_ID FOREIGN KEY (CUSTOMER\_ID) REFERENCES CUSTOMER\_TABLE (CUSTOMER\_ID),

CONSTRAINT FK\_STAFF\_ID FOREIGN KEY (STAFF\_ID) REFERENCES STAFF\_TABLE (STAFF\_ID)

);

CREATE TABLE TREATMENT\_TABLE (

TREATMENT\_ID VARCHAR(8),

ANIMAL\_ID VARCHAR(4),

DISEASE\_ID VARCHAR(9),

BILL\_ID VARCHAR(9),

VET\_ID VARCHAR(4),

MED\_ID VARCHAR(9),

CONSTRAINT PK\_TREATMENT\_ID PRIMARY KEY (TREATMENT\_ID),

CONSTRAINT FK\_ANIMAL\_ID FOREIGN KEY (ANIMAL\_ID) REFERENCES ANIMAL\_TABLE (ANIMAL\_ID),

CONSTRAINT FK\_DISEASE\_ID FOREIGN KEY (DISEASE\_ID) REFERENCES DISEASE\_TABLE (DISEASE\_ID),

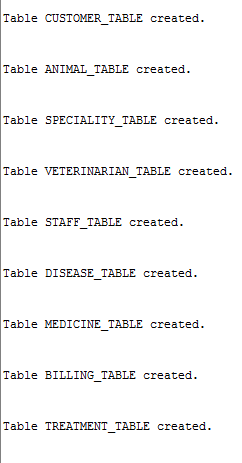
CONSTRAINT FK\_BILL\_ID FOREIGN KEY (BILL\_ID) REFERENCES BILLING\_TABLE (BILL\_ID),

CONSTRAINT FK\_VET\_ID FOREIGN KEY (VET\_ID) REFERENCES VETERINARIAN\_TABLE (VET\_ID),

CONSTRAINT FK\_MED\_ID FOREIGN KEY (MED\_ID) REFERENCES MEDICINE\_TABLE (MED\_ID)

);

OUTPUT:



INSERT ALL

INTO CUSTOMER\_TABLE VALUES ('10111', 'Pam', 'Willis', '041 123 4587', 'pw@ymail.com')

INTO CUSTOMER\_TABLE VALUES ('10112', 'Henry', 'Jones', '021 457 2587', 'hjon@isat.com')

INTO CUSTOMER\_TABLE VALUES ('10113', 'Alex', 'Smith', '011 125 5874', 'asm@conz.co.za')

INTO CUSTOMER\_TABLE VALUES ('10114', 'Wayne', 'Willard', '011 125 8548 ', 'ww@ymail.com')

INTO CUSTOMER\_TABLE VALUES ('10115', 'Sisanda', 'Bitterhout', '051 545 8987', 'sbitr@hmail.com')

SELECT 1

FROM DUAL;

--Animal Inserts

INSERT ALL

INTO ANIMAL\_TABLE VALUES ('A50', 'Horse', 'American Quarter', 'Tan')

INTO ANIMAL\_TABLE VALUES ('A51', 'Dog', 'Bichon Frise', 'White')

INTO ANIMAL\_TABLE VALUES ('A52', 'Bird', 'African Grey', 'Grey')

INTO ANIMAL\_TABLE VALUES ('A53', 'Horse', 'Friesian Horse', 'Black')

INTO ANIMAL\_TABLE VALUES ('A54', 'Cat', 'American Shorthair', 'Tan')

INTO ANIMAL\_TABLE VALUES ('A55', 'Horse', 'Appaloosa', 'Black')

INTO ANIMAL\_TABLE VALUES ('A56', 'Dog', 'Clumber Spaniel', 'White')

INTO ANIMAL\_TABLE VALUES ('A57', 'Bird', 'Conure', 'Green')

INTO ANIMAL\_TABLE VALUES ('A58', 'Bird', 'Macaw', 'Blue and Yellow')

INTO ANIMAL\_TABLE VALUES ('A59', 'Cat', 'Bombay', ' Black')

SELECT 1

FROM DUAL;

--Speciality Inserts

INSERT ALL

INTO SPECIALITY\_TABLE VALUES (1, 'Equus caballus')

INTO SPECIALITY\_TABLE VALUES (2, 'Canis lupus')

INTO SPECIALITY\_TABLE VALUES (3, 'Felis Catus')

INTO SPECIALITY\_TABLE VALUES (4, 'Aves')

SELECT 1

FROM DUAL;

--Veterinarian Inserts

INSERT ALL

INTO VETERINARIAN\_TABLE VALUES ('555', 'Sam', 'Anderson', '086 1248854', 'san@ymail.com', 1)

INTO VETERINARIAN\_TABLE VALUES ('556', 'Eric', 'Johnson', '072 5486587', 'ej@isat.com', 1)

INTO VETERINARIAN\_TABLE VALUES ('557', 'Jeffery', 'Tims', '083 1545854', 'jt@ymail.com', 3)

INTO VETERINARIAN\_TABLE VALUES ('558', 'Simon', 'Gumede', '081 5855983', 'sge@isat.com', 2)

INTO VETERINARIAN\_TABLE VALUES ('559', 'Maxi', 'Aneshia', '076 5858547', 'max@hmail.com', 4)

SELECT 1

FROM DUAL;

--Staff Inserts

INSERT ALL

INTO STAFF\_TABLE VALUES ('111', 'Jenny', 'Loots', 'Receptionist', '086 129 1198')

INTO STAFF\_TABLE VALUES ('112', 'Monique', 'Van Eyck', 'Assistant', '087 545 5851')

SELECT 1

FROM DUAL;

--Disease Inserts

INSERT ALL

INTO DISEASE\_TABLE VALUES ('dis\_101', 'Cerebellar abiotrophy')

INTO DISEASE\_TABLE VALUES ('dis\_102', 'Salmonellosis')

INTO DISEASE\_TABLE VALUES ('dis\_103', 'Proventricular Dilatation')

INTO DISEASE\_TABLE VALUES ('dis\_104', 'Candidiasis')

INTO DISEASE\_TABLE VALUES ('dis\_105', 'Parvo')

INTO DISEASE\_TABLE VALUES ('dis\_106', 'Periodontis')

SELECT 1

FROM DUAL;

--Medicine Inserts

INSERT ALL

INTO MEDICINE\_TABLE VALUES ('med\_101', 'Anti-Inflammatory', 180.95, '25 March 2018')

INTO MEDICINE\_TABLE VALUES ('med\_102', 'Electrolyte Replacement', 298.75, '13 April 2018')

INTO MEDICINE\_TABLE VALUES ('med\_103', 'COX-2 Inhibitors', 307.09, '8 February 2018')

INTO MEDICINE\_TABLE VALUES ('med\_104', 'Antifungal Tablets', 55.98, '19 March 2018')

INTO MEDICINE\_TABLE VALUES ('med\_105', 'Colostrum', 567.09, '17 January 2018')

INTO MEDICINE\_TABLE VALUES ('med\_106', 'Antiparasitic Tablets', 290.98, '23 October 2018')

INTO MEDICINE\_TABLE VALUES ('med\_107', 'Compax oral hygiene tablets', 125.89, '25 June 2018')

SELECT 1

FROM DUAL;

--Billing Inserts

INSERT ALL

INTO BILLING\_TABLE VALUES ('bil\_123', '10111', '111', 5)

INTO BILLING\_TABLE VALUES ('bil\_124', '10115', '111', 3)

INTO BILLING\_TABLE VALUES ('bil\_125', '10113', '112', 7)

INTO BILLING\_TABLE VALUES ('bil\_126', '10111', '111', 8)

INTO BILLING\_TABLE VALUES ('bil\_127', '10112', '112', 10)

SELECT 1

FROM DUAL;

--Treatment Inserts

INSERT ALL

INTO TREATMENT\_TABLE VALUES ('t\_111', 'A50', 'dis\_101', 'bil\_124', '555', 'med\_104')

INTO TREATMENT\_TABLE VALUES ('t\_112', 'A55', 'dis\_101', 'bil\_127', '555', 'med\_101')

INTO TREATMENT\_TABLE VALUES ('t\_113', 'A51', 'dis\_105', 'bil\_123', '558', 'med\_103')

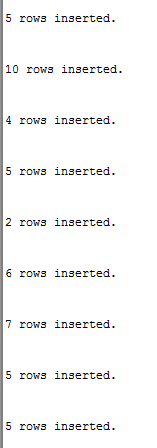
INTO TREATMENT\_TABLE VALUES ('t\_114', 'A57', 'dis\_104', 'bil\_126', '559', 'med\_103')

INTO TREATMENT\_TABLE VALUES ('t\_115', 'A51', 'dis\_105', 'bil\_125', '558', 'med\_104')

SELECT 1

FROM DUAL;

OUTPUT:



Question 1

--Question 1

SET SERVEROUTPUT ON

DECLARE

DISPLAY\_NAME VARCHAR(50);

DISPLAY\_BILL BILLING\_TABLE.BILL\_ID%TYPE;

DISPLAY\_MEDICINE MEDICINE\_TABLE.MED\_NAME%TYPE;

DISPLAY\_PRICE MEDICINE\_TABLE.MED\_PRICE%TYPE;

CURSOR MedicineCursor IS SELECT

FIRST\_NAME || ',' || SURNAME AS NAME,

BILLING\_TABLE.BILL\_ID,

MEDICINE\_TABLE.MED\_NAME,

MED\_PRICE

FROM TREATMENT\_TABLE

, BILLING\_TABLE

, CUSTOMER\_TABLE

, MEDICINE\_TABLE

WHERE BILLING\_TABLE.BILL\_ID = TREATMENT\_TABLE.BILL\_ID

AND CUSTOMER\_TABLE.CUSTOMER\_ID = BILLING\_TABLE.CUSTOMER\_ID

AND MEDICINE\_TABLE.MED\_ID = TREATMENT\_TABLE.MED\_ID

AND MEDICINE\_TABLE.MED\_PRICE > 200;

BEGIN

OPEN MedicineCursor;

LOOP

FETCH MedicineCursor INTO DISPLAY\_NAME, DISPLAY\_BILL, DISPLAY\_MEDICINE, DISPLAY\_PRICE;

EXIT WHEN MedicineCursor%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE('CUSTOMER: ' || DISPLAY\_NAME);

DBMS\_OUTPUT.PUT\_LINE('BILL ID: ' || DISPLAY\_BILL);

DBMS\_OUTPUT.PUT\_LINE('MEDICINE: ' || DISPLAY\_MEDICINE);

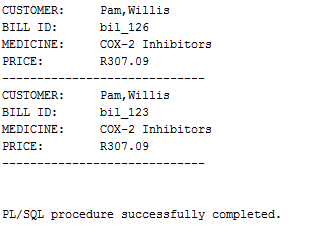
DBMS\_OUTPUT.PUT\_LINE('PRICE: R' || DISPLAY\_PRICE);

DBMS\_OUTPUT.PUT\_LINE('-----------------------------');

END LOOP;

END;

OUTPUT:



Question 2

--Question 2

DECLARE

DISPLAY\_CUSTOMER CUSTOMER\_TABLE.CUSTOMER\_ID%TYPE;

DISPLAY\_DISEASE DISEASE\_TABLE.DISEASE\_NAME%TYPE;

DISPLAY\_TYPE ANIMAL\_TABLE.ANIMAL\_TYPE%TYPE;

DISPLAY\_MEDICINE MEDICINE\_TABLE.MED\_NAME%TYPE;

DISPLAY\_PRICE MEDICINE\_TABLE.MED\_PRICE%TYPE;

DISPLAY\_DISCOUNT MEDICINE\_TABLE.MED\_PRICE%TYPE;

CURSOR DiscountCursor IS SELECT

CUSTOMER\_TABLE.CUSTOMER\_ID,

DISEASE\_TABLE.DISEASE\_NAME,

ANIMAL\_TABLE.ANIMAL\_TYPE,

MEDICINE\_TABLE.MED\_NAME,

MEDICINE\_TABLE.MED\_PRICE

FROM

TREATMENT\_TABLE

, BILLING\_TABLE

, CUSTOMER\_TABLE

, MEDICINE\_TABLE

, DISEASE\_TABLE

, ANIMAL\_TABLE

WHERE BILLING\_TABLE.BILL\_ID = TREATMENT\_TABLE.BILL\_ID

AND CUSTOMER\_TABLE.CUSTOMER\_ID = BILLING\_TABLE.CUSTOMER\_ID

AND MEDICINE\_TABLE.MED\_ID = TREATMENT\_TABLE.MED\_ID

AND DISEASE\_TABLE.DISEASE\_ID = TREATMENT\_TABLE.DISEASE\_ID

AND ANIMAL\_TABLE.ANIMAL\_ID = TREATMENT\_TABLE.ANIMAL\_ID

ORDER BY CUSTOMER\_ID ASC;

BEGIN

OPEN DiscountCursor;

LOOP

FETCH DiscountCursor INTO DISPLAY\_CUSTOMER

, DISPLAY\_DISEASE

, DISPLAY\_TYPE

, DISPLAY\_MEDICINE

, DISPLAY\_PRICE;

EXIT WHEN DiscountCursor%NOTFOUND;

DISPLAY\_DISCOUNT := 0;

IF DISPLAY\_TYPE LIKE 'Horse'

THEN

DISPLAY\_DISCOUNT := ROUND(DISPLAY\_PRICE - (DISPLAY\_PRICE \* 0.15), 2);

END IF;

DBMS\_OUTPUT.PUT\_LINE('CUSTOMER ID: ' || DISPLAY\_CUSTOMER);

DBMS\_OUTPUT.PUT\_LINE('DISEASE: ' || DISPLAY\_DISEASE);

DBMS\_OUTPUT.PUT\_LINE('Animal Type: ' || DISPLAY\_TYPE);

DBMS\_OUTPUT.PUT\_LINE('MEDICINE: ' || DISPLAY\_MEDICINE);

DBMS\_OUTPUT.PUT\_LINE('PRICE: R' || DISPLAY\_PRICE);

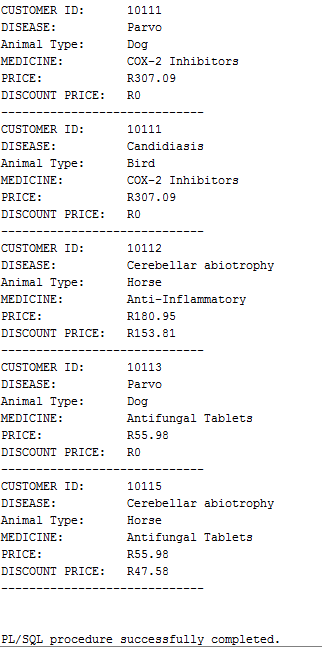
DBMS\_OUTPUT.PUT\_LINE('DISCOUNT PRICE: R' || DISPLAY\_DISCOUNT);

DBMS\_OUTPUT.PUT\_LINE('-----------------------------');

END LOOP;

END;

OUTPUT:



Question 3

--Question 3

DECLARE

DISPLAY\_VET\_ID VETERINARIAN\_TABLE.VET\_ID%TYPE;

DISPLAY\_SPECIALITY SPECIALITY\_TABLE.SPECIALITY%TYPE;

DISPLAY\_ANIMAL\_TYPE ANIMAL\_TABLE.ANIMAL\_TYPE%TYPE;

DISPLAY\_DISEASE DISEASE\_TABLE.DISEASE\_NAME%TYPE;

DISPLAY\_MEDICINE MEDICINE\_TABLE.MED\_NAME%TYPE;

DISPLAY\_STAFF\_ID STAFF\_TABLE.STAFF\_ID%TYPE;

DISPLAY\_CUSTOMER\_ID CUSTOMER\_TABLE.CUSTOMER\_ID%TYPE;

DISPLAY\_BILLING\_ID BILLING\_TABLE.BILL\_ID%TYPE;

CURSOR TreatmentCursor IS SELECT

TREATMENT\_TABLE.VET\_ID

, SPECIALITY\_TABLE.SPECIALITY

, ANIMAL\_TABLE.ANIMAL\_TYPE

, DISEASE\_TABLE.DISEASE\_NAME

, MEDICINE\_TABLE.MED\_NAME

, BILLING\_TABLE.STAFF\_ID

, CUSTOMER\_TABLE.CUSTOMER\_ID

, BILLING\_TABLE.BILL\_ID

FROM TREATMENT\_TABLE

, BILLING\_TABLE

, CUSTOMER\_TABLE

, MEDICINE\_TABLE

, DISEASE\_TABLE

, ANIMAL\_TABLE

, VETERINARIAN\_TABLE

, SPECIALITY\_TABLE

WHERE BILLING\_TABLE.BILL\_ID = TREATMENT\_TABLE.BILL\_ID

AND CUSTOMER\_TABLE.CUSTOMER\_ID = BILLING\_TABLE.CUSTOMER\_ID

AND MEDICINE\_TABLE.MED\_ID = TREATMENT\_TABLE.MED\_ID

AND DISEASE\_TABLE.DISEASE\_ID = TREATMENT\_TABLE.DISEASE\_ID

AND ANIMAL\_TABLE.ANIMAL\_ID = TREATMENT\_TABLE.ANIMAL\_ID

AND VETERINARIAN\_TABLE.VET\_ID = TREATMENT\_TABLE.VET\_ID

AND SPECIALITY\_TABLE.SPECIALTY\_ID = VETERINARIAN\_TABLE.SPECIALITY\_ID

AND MEDICINE\_TABLE.MED\_ID LIKE 'med\_101';

BEGIN

OPEN TreatmentCursor;

LOOP

FETCH TreatmentCursor INTO DISPLAY\_VET\_ID

, DISPLAY\_SPECIALITY

, DISPLAY\_ANIMAL\_TYPE

, DISPLAY\_DISEASE

, DISPLAY\_MEDICINE

, DISPLAY\_STAFF\_ID

, DISPLAY\_CUSTOMER\_ID

, DISPLAY\_BILLING\_ID;

EXIT WHEN TreatmentCursor%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE('VET ID: ' || DISPLAY\_VET\_ID);

DBMS\_OUTPUT.PUT\_LINE('SPECIALITY: ' || DISPLAY\_SPECIALITY);

DBMS\_OUTPUT.PUT\_LINE('Animal Type: ' || DISPLAY\_ANIMAL\_TYPE);

DBMS\_OUTPUT.PUT\_LINE('DISEASE: ' || DISPLAY\_DISEASE);

DBMS\_OUTPUT.PUT\_LINE('MEDICINE: ' || DISPLAY\_MEDICINE);

DBMS\_OUTPUT.PUT\_LINE('STAFF ID: ' || DISPLAY\_STAFF\_ID);

DBMS\_OUTPUT.PUT\_LINE('CUSTOMER ID: ' || DISPLAY\_CUSTOMER\_ID);

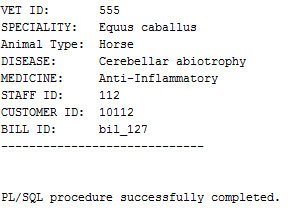
DBMS\_OUTPUT.PUT\_LINE('BILL ID: ' || DISPLAY\_BILLING\_ID);

DBMS\_OUTPUT.PUT\_LINE('-----------------------------');

END LOOP;

END;

OUTPUT:



Question 4

4.1 There are two types of Cursors. There are Implicit Cursors and Explicit Cursors. Implicit cursors are automatically created by oracle when a SQL statement that does not have an explicit cursor for the statement is executed. You often find implicit cursors in UPDATES and DELETES where the statement will find a selection of records to either change or delete the data. Explicit cursors are cursors that are programmatically defined. They are used to gain more control over the context area. When approaching an explicit cursor one must Declare variables, Open the cursor, Fetch, and then close the cursor. (Anon., n.d.)

4.2 A Data Dictionary is a file or a possible set of files that contains the metadata that will be used in a database. The dictionary will contain records about the various other objects that will be found within the database, such as data ownership, possible data relations to other objects, and the other data in the database. The metadata that the dictionary stores is the Names of the tables in the database and their owners, Names of all the indexes and columns to which the tables would relate to. Constraints on all the various tables. The most crucial thing that various relational database systems use the data dictionary for is, to access data within the database. (Technopedia, n.d.)

Question 5

The two types of blocks that a developer would make use of are Stored Procedures, and Anonymous code blocks. They are both very powerful and can both interact with one another. Store procedures allow the developer to create a block of code that can have whatever means of process that they desire, and allows them to reuse that said block of code throughout an application. The anonymous code block allows the user to call and use the stored procedures as often as they want. It also allows the developer to create new logic on the fly while incorporating other blocks of code in the application. (Oracle, n.d.)

Question 6

Higher Productivity- PL/SQL lets you write very small compact code for manipulating the data found in the database. PL/SQL also saves time on debugging and design by offering various features such as exception handling, encapsulation, object orientated data types, and data hiding.

Full Portability – Applications that are written in PL/SQL can run on any operating system, where the oracle database runs. You can also write portable program libraries and reuse them, in various environments.

Tight Security – PL/SQL stored procedures have the ability to move application code to the client server, where you can ensure its safety from tampering and viewing.

Access to Pre-defined packages – ORACLE has a range of product specific packages that is shipped with different versions of their systems. Each of these packages have different abilities.

Support for Object-Orientated Programming – In oracle you have the ability to make use of Object types which are very good for modeling data, and will reduce the cost and time required to build complex applications. It will also make your application more modular, maintainable and useable. (Oracle, n.d.)

Question 7

Make or replace procedure Student\_Count (course\_name INPUT tblResults.Course%TYPE, stud\_count SEND tblResults.Results%TYPE)

IS BEGIN

select count(r.results)

into stud\_count

from tblResults r

where r.Course = course\_name;

END;

set serveroutput on;

declaring

stud\_count tblResults.Results%OF\_TYPE;

begin

Count\_Student('DISD3',stud\_count);

dbms\_output.put\_line('---------------------------');

dbms\_output.put\_line('Number of students are: ' || student\_count);

end;

P.T.O

CORRECT VERSION

SET SERVEROUTPUT ON;

CREATE OR REPLACE PROCEDURE Student\_Count (course\_name IN tblResults.Course%TYPE, stud\_count OUT tblResults.Results%TYPE)

IS

BEGIN

SELECT COUNT(r.results) INTO STUD\_COUNT

FROM tblResults r

WHERE r.COURSE LIKE course\_name;

END;

DECLARE

stud\_count tblResults.Results%TYPE;

BEGIN

Student\_Count('DISD3',stud\_count);

dbms\_output.put\_line('---------------------------');

dbms\_output.put\_line('Number of students are: ' || stud\_count);

end;

Question 8

In Oracle every Constant Parameter and Variable has a data type. The data type determines how the data is stored i.e. its format, the constraints on that data, valid range of values for that data, and what kind of operations can be performed on that data. The basic data types that are used more frequently within databases are : NUMBER data types – Dec, Decimal or Numeric (Fixed-point Number with a maximum precision of 38 decimal digits), Double Precision or Float (Floating-point Number with a maximum precision of 128 binary digits), INT, Integer or SmallInt (Integer with maximum precision of 38 decimal digits). All of these number data types are used in different situations, such as if you are only looking for a single digit number such as an id number, then you would make use of the Number datatype. And if you are looking for numbers that will need to store multiple digit numbers then you would make use of Float or Decimal. There are a range of character data types that are used to store ‘Strings’. You can store single character strings in a data type such as ‘CHAR’, or a string that can be a long sentence such as VARCHAR (maximum length of 32,767 bytes). All of these data types will also for different operations to be performed such an comparison operations between strings and numbers will be different.

Question 9

CREATE OR REPLACE VIEW medicine\_expiry(MED\_ID,MED\_NAME,ANIMAL\_ID,ANIMAL,VETERINARIAN,SPECIALITY,EXPIRY\_DATE)

AS

SELECT

MEDICINE\_TABLE.MED\_ID

,MEDICINE\_TABLE.MED\_NAME

,ANIMAL\_TABLE.ANIMAL\_ID

,ANIMAL\_TYPE,VETERINARIAN\_TABLE.FIRST\_NAME||','||VETERINARIAN\_TABLE.SURNAME

,SPECIALITY\_TABLE.SPECIALITY

,(MED\_EXPIRY\_DATE-5)

FROM TREATMENT\_TABLE,MEDICINE\_TABLE,ANIMAL\_TABLE,VETERINARIAN\_TABLE,SPECIALITY\_TABLE

WHERE MEDICINE\_TABLE.MED\_ID = TREATMENT\_TABLE.MED\_ID

AND ANIMAL\_TABLE.ANIMAL\_ID = TREATMENT\_TABLE.ANIMAL\_ID

AND VETERINARIAN\_TABLE.VET\_ID = TREATMENT\_TABLE.VET\_ID

AND SPECIALITY\_TABLE.SPECIALTY\_ID = VETERINARIAN\_TABLE.SPECIALITY\_ID

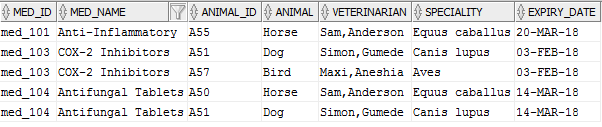
ORDER BY MED\_ID ASC;

OUTPUT:



SELECT \*

FROM medicine\_expiry;



Question 10

DECLARE

DISPLAY\_DISEASE DISEASE\_TABLE.DISEASE\_NAME%TYPE ;

DISPLAY\_ANIMAL\_TYPE ANIMAL\_TABLE.ANIMAL\_TYPE%TYPE;

DISPLAY\_NAME VETERINARIAN\_TABLE.FIRST\_NAME%TYPE;

CHECK\_COUNT NUMBER;

TREATMENT\_COUNT NUMBER;

DISPLAY\_LIST VARCHAR(200);

CURSOR ReportCursor IS SELECT

ANIMAL\_TABLE.ANIMAL\_TYPE

, DISEASE\_TABLE.DISEASE\_NAME

, VETERINARIAN\_TABLE.FIRST\_NAME || ',' || VETERINARIAN\_TABLE.SURNAME

, COUNT(DISEASE\_NAME)

FROM TREATMENT\_TABLE

, MEDICINE\_TABLE

, DISEASE\_TABLE

, ANIMAL\_TABLE

, VETERINARIAN\_TABLE

WHERE MEDICINE\_TABLE.MED\_ID = TREATMENT\_TABLE.MED\_ID

AND DISEASE\_TABLE.DISEASE\_ID = TREATMENT\_TABLE.DISEASE\_ID

AND ANIMAL\_TABLE.ANIMAL\_ID = TREATMENT\_TABLE.ANIMAL\_ID

AND VETERINARIAN\_TABLE.VET\_ID = TREATMENT\_TABLE.VET\_ID

GROUP BY DISEASE\_NAME, ANIMAL\_TYPE, FIRST\_NAME, SURNAME;

BEGIN

OPEN ReportCursor;

TREATMENT\_COUNT := 0;

LOOP

FETCH ReportCursor INTO DISPLAY\_ANIMAL\_TYPE, DISPLAY\_DISEASE, DISPLAY\_NAME, CHECK\_COUNT;

EXIT WHEN ReportCursor%NOTFOUND;

IF CHECK\_COUNT > 1

THEN

DISPLAY\_LIST := DISPLAY\_LIST || DISPLAY\_DISEASE || ',';

END IF;

TREATMENT\_COUNT := TREATMENT\_COUNT +1;

DBMS\_OUTPUT.PUT\_LINE('TREATMENT DETAILS ' || TREATMENT\_COUNT);

DBMS\_OUTPUT.PUT\_LINE(' DISEASE: ' || DISPLAY\_DISEASE);

DBMS\_OUTPUT.PUT\_LINE(' ANIMAL: ' || DISPLAY\_ANIMAL\_TYPE);

DBMS\_OUTPUT.PUT\_LINE(' VET: ' || DISPLAY\_NAME);

DBMS\_OUTPUT.PUT\_LINE('---------------------------------');

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('REPORT DETAILS');

DBMS\_OUTPUT.PUT\_LINE('---------------------------------');

DBMS\_OUTPUT.PUT\_LINE('DISEASE COUNT: '|| TREATMENT\_COUNT);

DBMS\_OUTPUT.PUT\_LINE('ANIMAL COUNT: '|| TREATMENT\_COUNT);

DBMS\_OUTPUT.PUT\_LINE('VET COUNT: '|| TREATMENT\_COUNT);

DBMS\_OUTPUT.PUT\_LINE('---------------------------------');

IF DISPLAY\_LIST IS NOT NULL

THEN

DBMS\_OUTPUT.PUT\_LINE('DISEASE WARNING');

DBMS\_OUTPUT.PUT\_LINE('---------------------------------');

DBMS\_OUTPUT.PUT\_LINE(' '||DISPLAY\_LIST);

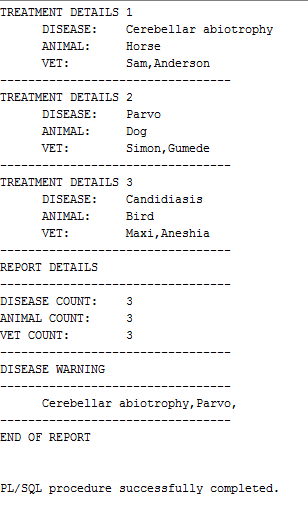
DBMS\_OUTPUT.PUT\_LINE('---------------------------------');

DBMS\_OUTPUT.PUT\_LINE('END OF REPORT');

END IF;

END;

OUTPUT:



Question 11

A continue statement will exit the current iteration of a loop. It will either happen conditionally or unconditionally. It transfers control to the next iteration of the loop or it can continue to another loop.

# Bibliography

Anon., n.d. *Implicit & Explicit Cursors.* [Online]   
Available at: https://www.tutorialspoint.com/plsql/plsql\_cursors.htm

Oracle, n.d. *Oracle Docs Stored Procedures.* [Online]   
Available at: https://docs.oracle.com/cd/B25329\_01/doc/appdev.102/b25108/xedev\_programs.htm  
[Accessed 2017].

Technopedia, n.d. *Data Dictionary.* [Online]   
Available at: https://www.techopedia.com/definition/27752/data-dictionary  
[Accessed 22 May 2017].