CPRG 307-E - Database Programming

Assignment 2: Develop and Test a Coded Solution

Prepared by

Gabriel Ira Siwa

Lulubelle Fontelo

Mitzi Vera Escartin

Alessandra Nicole Claur

Prepared for

Nicole Berard

Date: March 21, 2025

School for Advanced Digital Technology

Syntax Error Identification, Explanation, and Correction:

<u>Syntax Error 1</u>

Code Section in Error (Line 47)

```
41        ELSIF (r_gggs.process_type = k_change) THEN
42          UPDATE gggs_customer
43             SET province = DECODE(r_gggs.column2, k_no_change_char, province, r_gggs.column2),
44                 first_name = DECODE(r_gggs.column3, k_no_change_char, first_name, r_gggs.column3),
45                 last_name = DECODE(r_gggs.column4, k_no_change_char, last_name, r_gggs.column4),
46                 city = DECODE(r_gggs.column5, k_no_change_char, city, r_gggs.column5),
47                 phone_number = NVL(r_gggs.column6, r_gggs.column6, phone_number)
48           WHERE name = r_gggs.column1;
```

Error Message Received

```
ORA-06550: line 47, column 33:
PL/SQL: ORA-00909: invalid number of arguments
```

Reason for error:

- The NVL function is trying to use three arguments, but the function only allows two arguments in PL/SQL.

Correction:

- The correct syntax is to provide only two arguments when using NVL.

```
41        ELSIF (r_gggs.process_type = k_change) THEN
42          UPDATE gggs_customer
43             SET province = DECODE(r_gggs.column2, k_no_change_char, province, r_gggs.column2),
44                 first_name = DECODE(r_gggs.column3, k_no_change_char, first_name, r_gggs.column3),
45                 last_name = DECODE(r_gggs.column4, k_no_change_char, last_name, r_gggs.column4),
46                 city = DECODE(r_gggs.column5, k_no_change_char, city, r_gggs.column5),
47                 phone_number = NVL(r_gggs.column6, phone_number)
48           WHERE name = r_gggs.column1;
```

Syntax Error 2

Code Section in Error: (Line 72)

```
65        ELSIF (r_gggs.process_type = k_change) THEN
66          UPDATE gggs_vendor
67             SET description = DECODE(r_gggs.column2, k_no_change_char, description, r_gggs.column2),
68                 contact_first_name = DECODE(r_gggs.column3, k_no_change_char, contact_first_name, r_gggs.column3),
69                 contact_last_name = DECODE(r_gggs.column4, k_no_change_char, contact_last_name, r_gggs.column4),
70                 contact_phone_number = NVL2(r_gggs.column6, r_gggs.column6, contact_phone_number)
71             WHERE name = r_gggs.column1
72        ELSE
```

Error Message Received:

```
ORA-06550: line 72, column 9:
PL/SQL: ORA-00933: SQL command not properly ended
```

Reason for Error:

- The SQL statement is missing a semicolon (;), causing PL/SQL to not recognize the end of the statement.

Correction:

- Add a semicolon at the end of the Update statement.

```
65        ELSIF (r_gggs.process_type = k_change) THEN
66          UPDATE gggs_vendor
67             SET description = DECODE(r_gggs.column2, k_no_change_char, description, r_gggs.column2),
68                 contact_first_name = DECODE(r_gggs.column3, k_no_change_char, contact_first_name, r_gggs.column3),
69                 contact_last_name = DECODE(r_gggs.column4, k_no_change_char, contact_last_name, r_gggs.column4),
70                 contact_phone_number = NVL2(r_gggs.column6, r_gggs.column6, contact_phone_number)
71             WHERE name = r_gggs.column1;
72        ELSE
```

Syntax Error 3

Code Section in Error: (Line 121)

```
107         ELSIF (r_gggs.process_type = k_status) THEN
108           UPDATE gggs_stock
109              SET status = r_gggs.column2
110            WHERE name = r_gggs.column1;
111
112         ELSE IF (r_gggs.process_type = k_change) THEN
113           UPDATE gggs_stock
114              SET description = DECODE(r_gggs.column4, k_no_change_char, description, r_gggs.column4),
115                  price = NVL2(r_gggs.column7, r_gggs.column7, price),
116                  no_in_stock = NVL2(r_gggs.column8, (no_in_stock - r_gggs.column8), no_in_stock)
117            WHERE name = r_gggs.column1;
```

Error Message Received:

```
ORA-06550: line 121, column 4:
PLS-00103: Encountered the symbol "ELSE" when expecting one of the following:
```

Reason for Error:

- PL/SQL does not support ELSE IF. The correct syntax is ELSIF. The Else in line 122 is not connected to a valid If, leading to the error PLS-00103.

Correction:

- Change ELSE IF to ELSIF

```
112         ELSIF (r_gggs.process_type = k_change) THEN
113           UPDATE gggs_stock
114              SET description = DECODE(r_gggs.column4, k_no_change_char, description, r_gggs.column4)
115                  price = NVL2(r_gggs.column7, r_gggs.column7, price),
116                  no_in_stock = NVL2(r_gggs.column8, (no_in_stock - r_gggs.column8), no_in_stock)
117            WHERE name = r_gggs.column1;
118         ELSE
```

Syntax Error 4

Code Section in Error:

```
60          ELSIF (r_gggs.process_type = k_stats) THEN
61            UPDATE gggs_vendor
62               SET status = r_gggs.column2
63             WHERE name = r_gggs.column1;
64
65          ELSIF (r_gggs.process_type = k_change) THEN
```

Error Message Received:

```
Error report -
ORA-06550: line 60, column 38:
PLS-00201: identifier 'K_STATS' must be declared
```

Reason for Error:

- k_stats should be k_status – k_stats is not declared

Correction:

- Change k_stats to k_status

```
60          ELSIF (r_gggs.process_type = k_status) THEN
61            UPDATE gggs_vendor
62               SET status = r_gggs.column2
63             WHERE name = r_gggs.column1;
```

Syntax Error 5

Code Section in Error: (Line 147)

```
141
142         COMMIT;
143
144
145   END LOOP;
146
147 END;
148 /
```

Error Message Received:

```
Error report -
ORA-06550: line 147, column 4:
PLS-00103: Encountered the symbol ";" when expecting one of the following:
```

Reason for Error:

- The error occurred because there was a missing `END;` inside the `BEGIN ... END;` block before `END LOOP;`. If a `BEGIN` block is opened inside the loop, it must be properly closed before `END LOOP;`.

Correction:

- Add END to close the Exception block before the end loop

```
131
132     EXCEPTION
133       WHEN OTHERS THEN
134         ROLLBACK;
135
136         v_message := SQLERRM;
137
138         INSERT INTO  gggs_error_log_table
139         VALUES
140          (r_gggs.data_type, r_gggs.process_type, v_message);
141
142         COMMIT;
143     END; -- FIX: This END should close the EXCEPTION block before END LOOP
144
145
146     END LOOP;
147
148 END;
149 /
```

Logical Error Identification, Explanation, and Correction:

Logical Error 1

Expected Result:

- There should be 6 data entries for table gggs_stock

STOCKID	CATEGORYID	SUPPLIERID	NAME	DESCRIPTION	PRICE	NO_IN_STOCK	STATUS
1	2	2	GBRGFF	Gladiolus	7.49	23	A
2	2	2	GBRDG	Dicentra Garden	29.99	42	A
3	1	4	GSDB	Broccoli	2.49	133	A
4	1	4	GSDE	Eggplant	5.99	3	I
5	1	5	GSOKCFQ	Crane Feather Queen Orname	4.49	100	A
6	5	2	FBDASP	Dahlia Special	6.99	55	A

Actual Result:

- Only 4 rows of data entries are showing

	STOCKID	CATEGORYID	SUPPLIERID	NAME	DESCRIPTION	PRICE	NO_IN_STOCK	STATUS
1	1	2	2	GBRGFF	Gladiolus	7.49	23	A
2	2	2	2	GBRDG	Dicentra Garden	29.99	42	A
3	3	1	4	GSDB	Broccoli	2.49	133	A
4	4	1	4	GSDE	Eggplant	5.99	3	I

Code Section in Error:

```
    ELSIF (r_gggs.data_type = k_stock) THEN
      IF (r_gggs.process_type = k_new) THEN
        SELECT categoryID
          INTO v_name1
          FROM gggs_category
         WHERE name = r_gggs.column1;

        SELECT vendorID
          INTO v_name2
          FROM gggs_vendor
         WHERE name = r_gggs.column3;

          INSERT INTO gggs_stock
          VALUES (gggs_stock_seq.NEXTVAL, v_name1, v_name2, r_gggs.column3,
                  r_gggs.column4, r_gggs.column7, r_gggs.column8, k_active_status);
```

Reason for Error:

- If r_gggs.column2 does not contain the correct name value for gggs_vendor, the SELECT vendorID statement would fail, resulting in an error or missing data.

Correction:

- Changed the name in WHERE clause, from r_gggs.column3 to r_gggs.column2

```
SELECT vendorID
  INTO v_name2
  FROM gggs_vendor
 WHERE name = r_gggs.column2;
```

Logical Error 2

Expected Result:

- no_in_stock of GSDB Broccoli in gggs_stock table should be 113

Actual Result:

- no_in_stock showing is 67

	STOCKID	CATEGORYID	SUPPLIERID	NAME	DESCRIPTION	PRICE	NO_IN_STOCK	STATUS
1	1	2	2	GBRGFF	Gladiolus	7.49	23	A
2	2	2	2	GBRDG	Dicentra Garden	29.99	42	A
3	3	1	4	GSDB	Broccoli	2.49	67	A

Code Section in Error:

```
        ELSIF (r_gggs.process_type = k_change) THEN
          UPDATE gggs_stock
             SET description = DECODE(r_gggs.column4, k_no_change_char, description, r_gggs.column4),
                 price = NVL2(r_gggs.column7, r_gggs.column7, price),
                 no_in_stock = NVL2(r_gggs.column8, (no_in_stock - r_gggs.column8), no_in_stock)
           WHERE name = r_gggs.column1;
```

Reason for Error:

- Incorrect arithmetic operation in the UPDATE statement for no_in_stock

Correction:

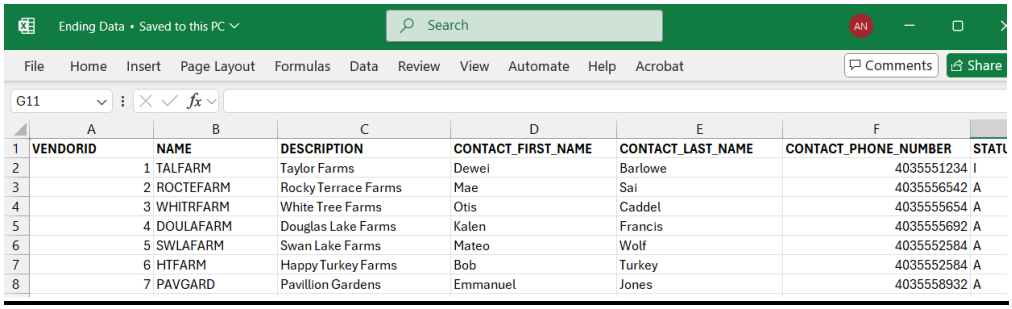
- Changed the operation from subtraction to addition

```
UPDATE gggs_stock
   SET description = DECODE(r_gggs.column4, k_no_change_char, description, r_gggs.column4),
       price = NVL2(r_gggs.column7, r_gggs.column7, price),
       no_in_stock = NVL2(r_gggs.column8, (no_in_stock + r_gggs.column8), no_in_stock)
 WHERE name = r_gggs.column1;
```

Logical Error 3

Expected Result:

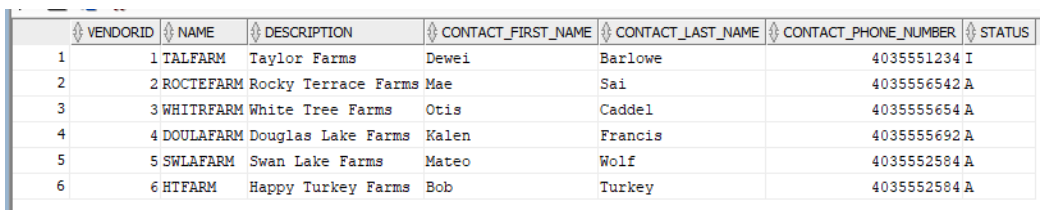
- There should be 7 rows of data entry in table gggs_vendor



VENDORID	NAME	DESCRIPTION	CONTACT_FIRST_NAME	CONTACT_LAST_NAME	CONTACT_PHONE_NUMBER	STATU
1	TALFARM	Taylor Farms	Dewei	Barlowe	4035551234	I
2	ROCTEFARM	Rocky Terrace Farms	Mae	Sai	4035556542	A
3	WHITRFARM	White Tree Farms	Otis	Caddel	4035555654	A
4	DOULAFARM	Douglas Lake Farms	Kalen	Francis	4035555692	A
5	SWLAFARM	Swan Lake Farms	Mateo	Wolf	4035552584	A
6	HTFARM	Happy Turkey Farms	Bob	Turkey	4035552584	A
7	PAVGARD	Pavillion Gardens	Emmanuel	Jones	4035558932	A

Actual Result:

- Missing data entry in table gggs_vendor



	VENDORID	NAME	DESCRIPTION	CONTACT_FIRST_NAME	CONTACT_LAST_NAME	CONTACT_PHONE_NUMBER	STATUS
1	1	TALFARM	Taylor Farms	Dewei	Barlowe	4035551234	I
2	2	ROCTEFARM	Rocky Terrace Farms	Mae	Sai	4035556542	A
3	3	WHITRFARM	White Tree Farms	Otis	Caddel	4035555654	A
4	4	DOULAFARM	Douglas Lake Farms	Kalen	Francis	4035555692	A
5	5	SWLAFARM	Swan Lake Farms	Mateo	Wolf	4035552584	A
6	6	HTFARM	Happy Turkey Farms	Bob	Turkey	4035552584	A

Code Section in Error:

```
      ELSIF (r_gggs.data_type = k_vendor) THEN

        IF (r_gggs.process_type = k_new) THEN
          INSERT INTO gggs_vendor
          VALUES (gggs_vendor_seq.NEXTVAL, r_gggs.column1, r_gggs.column2, r_gggs.column3,
                  r_gggs.column4, r_gggs.column6, k_status);

```

Reason for Error:

- k_status was used instead of k_active_status when inserting data into gggs_vendor. k_status does not hold the correct active status value

Correction:

- Change k_status to k_active_status

```
IF (r_gggs.process_type = k_new) THEN
  INSERT INTO gggs_vendor
  VALUES (gggs_vendor_seq.NEXTVAL, r_gggs.column1, r_gggs.column2, r_gggs.column3,
          r_gggs.column4, r_gggs.column6, k_acive_status);
```

Logical Error 4

Expected Result:

- The error message for data_type = 'ST' and process_type = 'R' should be in the correct place.

A1 | DATA_TYPE

	A	B	C
1	DATA_TYPE	PROCESS_TYPE	ERROR_MESSAGE
2	CA	Q	ORA-20001: Q is not a valid process request for CA data
3	ST	R	ORA-20001: R is not a valid process request for ST data
4	CU	N	ORA-00001: unique constraint (CA_A193_PLSQL_T01.UK_GGGS_CUSTOMER_NAME) violated
5	CU	A	ORA-20001: A is not a valid process request for CU data
6	CO	S	ORA-20000: CO is not a valid type of data to process
7	VE	L	ORA-20001: L is not a valid process request for VE data
8			
9			

Actual Result:

- The error messages are switched.

```
DA P ERROR_MESSAGE
-- - ----------------------------------------------------------------------
CA Q ORA-20001: Q is not a valid process request for CA data
ST R ORA-20001: ST is not a valid process request for R data
CU N ORA-00001: unique constraint (SYSTEM.UK_GGGS_CUSTOMER_NAME) violated
CU A ORA-20001: A is not a valid process request for CU data
CO S ORA-20000: CO is not a valid type of data to process
VE L ORA-20001: L is not a valid process request for VE data

6 rows selected.
```

Code Section in Error:

```
ELSE
  RAISE_APPLICATION_ERROR(-20001, r_gggs.data_type || ' is not a valid process request for ' || r_gggs.process_type || ' data');
END IF;

ELSE
```

Reason for Error:

- is that the code is incorrectly switching the data_type and process_type in the error message.

  This causes the system to interpret the wrong data as being invalid for a given process.

Correction:

- Switch r_gggs.process_type with r_gggs.data_type.

```
ELSE
  RAISE_APPLICATION_ERROR(-20001, r_gggs.process_type || ' is not a valid process request for ' || r_gggs.data_type || ' data'); -- LOGIC FIX: change r_gggs.data_type to r.gggs_process.type
END IF;
```

Logical Error 5

Expected Result:

- The StockID should be numbered consecutively.

A1 | STOCKID

	A	B	C	D	E	F	G	H
1	STOCKID	CATEGORYID	SUPPLIERID	NAME	DESCRIPTION	PRICE	NO_IN_STOCK	STATUS
2	1	2	2	GBRGFF	Gladiolus	7.49	23	A
3	2	2	2	GBRDG	Dicentra Garden	29.99	42	A
4	3	1	4	GSDB	Broccoli	2.49	133	A
5	4	1	4	GSDE	Eggplant	5.99	3	I
6	5	1	5	GSOKCFQ	Crane Feather Queen Orname	4.49	100	A
7	6	5	2	FBDASP	Dahlia Special	6.99	55	A

Actual Result:

- The StockID is skipping the numbers 5-6-7-8 and not continuously

Script Output | Query Result

SQL | All Rows Fetched: 6 in 0.017 seconds

	STOCKID	CATEGORYID	SUPPLIERID	NAME	DESCRIPTION	PRICE	NO_IN_STOCK	STATUS
1	1	2	2	GBRGFF	Gladiolus	7.49	23	A
2	2	2	2	GBRDG	Dicentra Garden	29.99	42	A
3	3	1	4	GSDB	Broccoli	2.49	133	A
4	4	1	4	GSDE	Eggplant	5.99	3	I
5	9	1	5	GSOKCFQ	Crane Feather Queen Ornamental Kale	4.49	100	A
6	10	5	2	FBDASP	Dahlia Special	6.99	55	A

Code Section in Error:

```
  END IF;

ELSIF (r_gggs.data_type = k_stock) THEN
  IF (r_gggs.process_type = k_new) THEN

        SELECT categoryID
        INTO v_name1
        FROM gggs_category
        WHERE name = r_gggs.column1;

        SELECT vendorID
        INTO v_name2
        FROM gggs_vendor
        WHERE name = r_gggs.column2; -- LOGICAL FIX: changed r_gggs.column3 to r_gggs.column2

        INSERT INTO gggs_stock
        VALUES (gggs_stock_seq.NEXTVAL, v_name1, v_name2, r_gggs.column3,
                r_gggs.column4, r_gggs.column7, r_gggs.column8, k_active_status);
    END IF;
  END;

  ELSIF (r_gggs.process_type = k_status) THEN
```

Reason for Error

- The error occurs because the StockID is generated using. NEXTVAL, which automatically increments each time it's called. This can cause gaps in the StockID sequence.

Correction:

- uses a subquery to calculate the next stockID by finding the maximum value in the stockID column and adding 1 to it. This is intended to dynamically assign the next available ID.

```
INSERT INTO gggs_stock
  VALUES (
      (SELECT NVL(MAX(stockID), 0) + 1 FROM gggs_stock),  -- Logical Fix: Assign next available StockID
      v_name1, v_name2, r_gggs.column3, r_gggs.column4, r_gggs.column7, r_gggs.column8, k_active_status
      );
ELSIF (r_gggs.process_type = k_status) THEN
```

Additional Recommendation:

1. Code Readability and Maintainability
   - One of the improvements we can recommend is to improve the coding style to make it more readable. Currently, the script contains long nested structures which can in future be difficult to read and debug. One way to improve this is by adding comments for each block explaining the purpose of each code such as update or insert. Use of consistent indention is also much better for structure clarity. Consider using a CASE Statement instead of multiple IF-ELSIF conditions.
2. Error Handling
   - Although the script raises errors with RAISE_APPLICATION_ERROR, which is a database level error; perhaps we can use descriptive, user-friendly messages that explain the issue, implement a lookup table to store standard error messages, and use a custom function to return formatted error descriptions.