

Fair Use Notice:

The material used in this presentation i.e., pictures/graphs/text, etc. is solely intended for educational/teaching purpose, offered free of cost to the students for use under special circumstances of Online Education due to COVID-19 Lockdown situation and may include copyrighted material - the use of which may not have been specifically authorised by Copyright Owners. It's application constitutes Fair Use of any such copyrighted material as provided in globally accepted law of many countries. The contents of presentations are intended only for the attendees of the class being conducted by the presenter.

DATABASE SYSTEMS (SW215)

DATA MANIPULATION LANGUAGE

By : HIRA NOMAN

CATEGORIES OF SQL STATEMENTS

1. Data Definition Languages (DDL).
2. Data Query Languages (DQL).
3. Data Manipulation Languages (DML).
4. Data Control Languages (DCL).
5. Transaction Control Languages (TCL).

DATA MANIPULATION LANGUAGES (DML)

- DML Commands are used to modify the data within db objects.
- Data Manipulation Language (DML) statements are used for managing data in database.
- DML commands are not auto-committed. It means changes made by DML command are not permanent to database.
- Changes made by DML commands are temporary and can be rolled back, if not made permanent.
- Following are the commands included in this category:

1. INSERT

2 UPDATE

3. DELETE

INSERTING DATA INTO A TABLE

- User can insert the data / records into specified table using **INSERT INTO** statement.
- It is possible to write the **INSERT INTO** Statement using 3 Types:

Type1: Without using column name

- The form does not specify the column names where the data will be inserted, one can directly enter their values BUT all the column values should be entered otherwise it will throw an error.

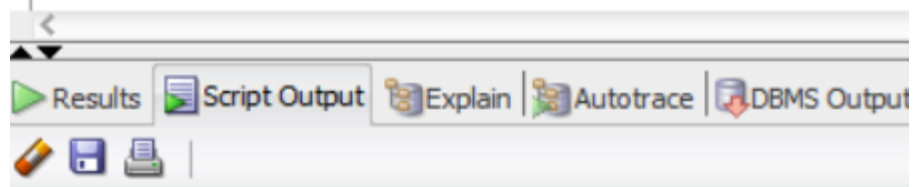
SYNTAX:

```
INSERT INTO table_name VALUES (Value1, Value2, Value3, Value4, Value5);
```

EXAMPLE:

```
INSERT INTO sw_students VALUES (101,'11-OCT-89');
```

```
desc sw_students
```



```
desc sw_students
```

Name	Null	Type
------	------	------

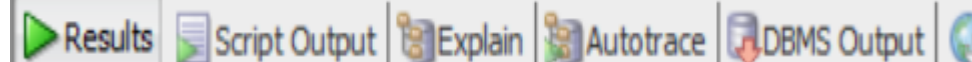
ST_ID		NUMBER(7)
S_DOB		DATE

2 rows selected

```
desc sw_students
```

```
INSERT INTO sw_students Values (101,'11-OCT-89');
```

```
select * from sw_students
```



Results:

	ST_ID	S_DOB
1	101	11-OCT-89

```
INSERT INTO sw_students Values ('11-OCT-89', 102);
```

Error encountered



An error was encountered performing the requested operation:

ORA-00932: inconsistent datatypes: expected DATE got NUMBER
00932. 00000 - "inconsistent datatypes: expected %s got %s"
*Cause:
*Action:
Vendor code 932Error at Line:2 Column:45

OK

```
INSERT INTO sw_students Values ('11-OCT-89');
```

Error encountered



An error was encountered performing the requested operation:

ORA-00947: not enough values
00947. 00000 - "not enough values"
*Cause:
*Action:
Vendor code 947Error at Line:2 Column:12

OK

Type 2: With using column names

This form specifies both the Column Names and the Values to be inserted.

One can also specify fewer columns and their values instead of ALL columns while using insert with column names.

SYNTAX

```
INSERT INTO table_name (Column1, Column2, Column3)
VALUES (Value1, Value2, Value3);
```

EXAMPLE:

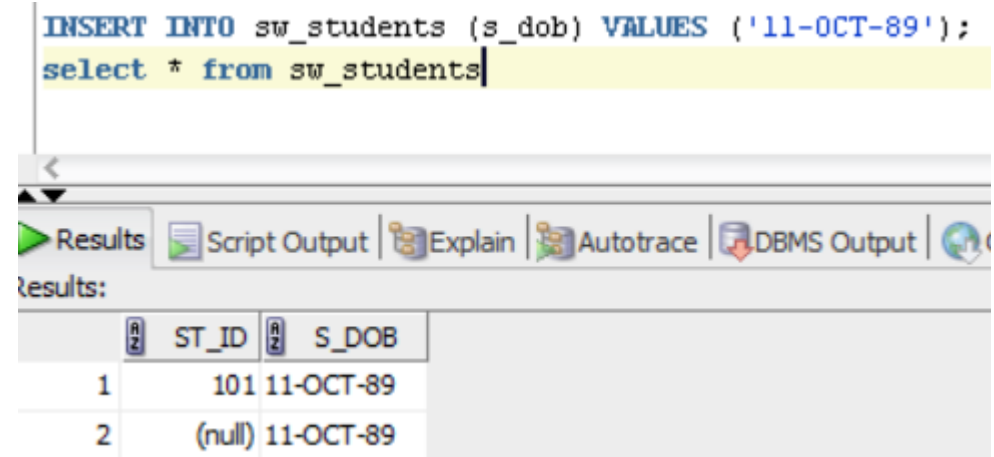
1.

```
INSERT INTO sw_students (st_id, s_dob)
VALUES (101,'11-OCT-89');
```
2.

```
INSERT INTO sw_students (s_dob)
VALUES ('11-OCT-89');
```
3.

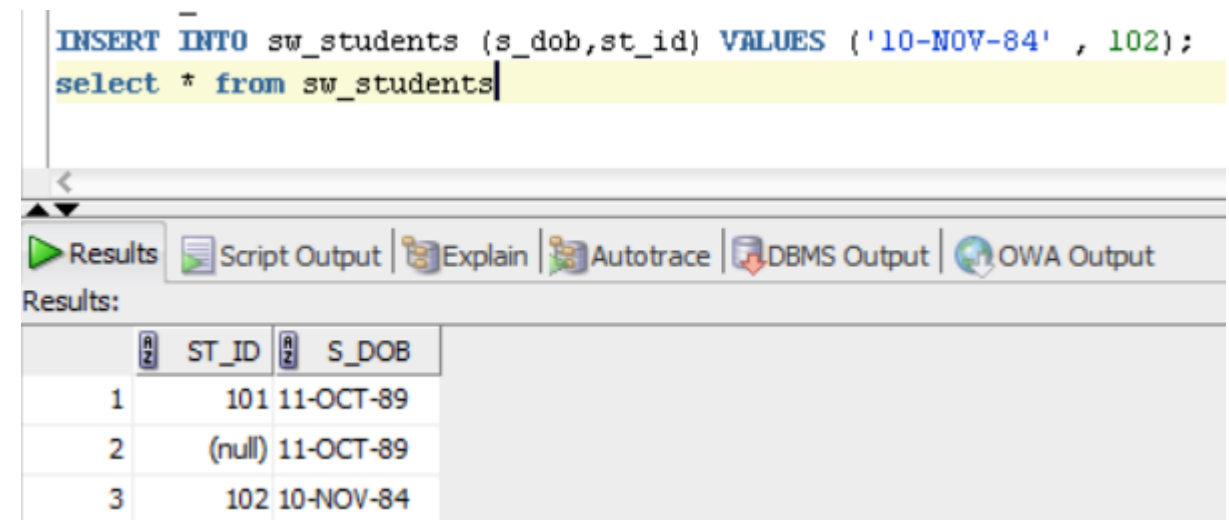
```
INSERT INTO sw_students (st_id,s_name)
VALUES ('10-OCT-89' , 102);
```

```
INSERT INTO sw_students (s_dob) VALUES ('11-OCT-89');
select * from sw_students
```



	ST_ID	S_DOB
1	101	11-OCT-89
2	(null)	11-OCT-89

```
INSERT INTO sw_students (s_dob,st_id) VALUES ('10-NOV-84' , 102);
select * from sw_students
```



	ST_ID	S_DOB
1	101	11-OCT-89
2	(null)	11-OCT-89
3	102	10-NOV-84

Type3: Insert Data through textbox

SYNTAX:

INSERT INTO Table_Name

VALUES (&Column1, &Column2, &Column3);

```
INSERT INTO sw_students VALUES (&st_id, &s_dob);  
select * from sw_students
```

Results Script Output Explain Autotrace DBMS Output

Results:

	ST_ID	S_DOB
1	101	11-OCT-89
2	(null)	11-OCT-89
3	102	10-NOV-84
4	103	18-FEB-21

INSERT INTO sw_students VALUES (&st_id, &s_dob);

Enter Substitution Variable

ST_ID:

103

OK Cancel

INSERT INTO sw_students VALUES (&st_id, &s_dob);

Enter Substitution Variable

S_DOB:

'18-FEB-21'

OK Cancel


```
INSERT INTO sw_students(S_DOB,ST_ID) VALUES (&DATE, &ID);
```

Enter Substitution Variable

DATE:

OK

Cancel

DBMS Output |  OWA O

Type 4 : Insert data into Table using SELECT statement

Inserting data to a table through a SELECT statement can be done in 2 ways:

1) If you are inserting data to all the columns, the Insert Statement can be written as:

- For this method to work its imperative that both the tables should have the same structure.

SYNTAX:

```
INSERT INTO Table_Name  
SELECT * FROM Table_Name ;
```

```
desc sw_students1
Name                                     Null      Type
-----
ST_ID                                  NUMBER(5)
S_NAME                                VARCHAR2(15)
S_DOB                                 DATE

3 rows selected
```

EXAMPLE:

```
INSERT INTO sw_students2
SELECT * FROM sw_students1;
```

```
desc sw_students2
```

Name	Null	Type
ST_ID		NUMBER(5)
S_NAME		VARCHAR2(15)
S_DOB		DATE

3 rows selected

```
select * from sw_students1
```

Results

Script Output

Explain

Autotrace

Results:

	ST_ID	S_NAME	S_DOB
1	1	abc	18-FEB-21
2	2	xyz	19-FEB-21
3	3	hij	17-FEB-21

```
INSERT INTO sw_students2 SELECT * FROM sw_students1 ;  
select * from sw_students2
```

Results

Script Output

Explain

Autotrace

DBMS Output

Results:

	ST_ID	S_NAME	S_DOB
1	1	abc	18-FEB-21
2	2	xyz	19-FEB-21
3	3	hij	17-FEB-21

```
desc sw_students
```

Name	Null	Type
ST_ID		NUMBER(7)
S_DOB		DATE
S_NAME		CHAR(15)

3 rows selected

```
desc sw_students1
```

Name	Null	Type
ST_ID		NUMBER(5)
S_NAME		VARCHAR2(15)
S_DOB		DATE

3 rows selected

```
INSERT INTO sw_students  
SELECT * FROM sw_students1
```

Error encountered



An error was encountered performing the requested operation:

ORA-01858: a non-numeric character was found where a numeric was expected

01858. 00000 - "a non-numeric character was found where a numeric was expected"

*Cause: The input data to be converted using a date format model was incorrect. The input data did not contain a number where a number was required by the format model.

*Action: Fix the input data or the date format model to make sure the elements match in number and type. Then retry the operation.

Vendor code 1858Error at Line:7

OK

2) If you are inserting data to specified columns, the Insert Statement can be written as:

SYNTAX:

INSERT INTO Table_Name [(Column1, Column2,...ColumnN)]

SELECT Column1, Column2,...ColumnN FROM Table_Name [WHERE condition] ;

```
INSERT INTO sw_students2 (st_id) SELECT st_id FROM sw_students1 ;
select * from sw_students2
```

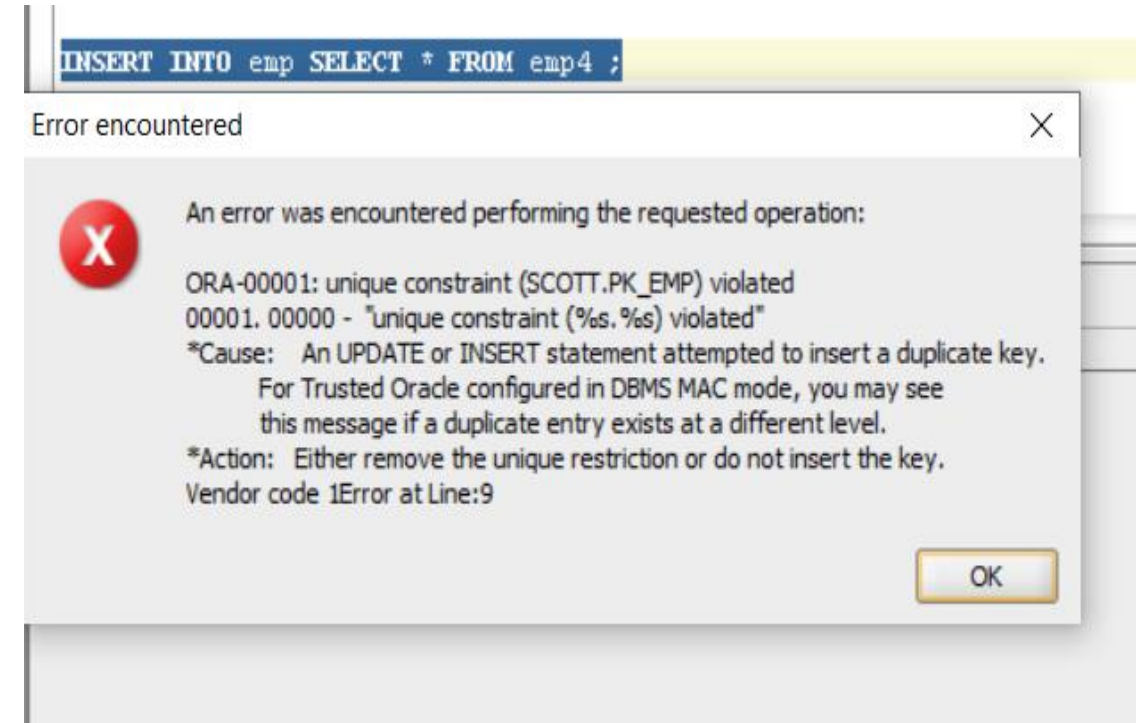
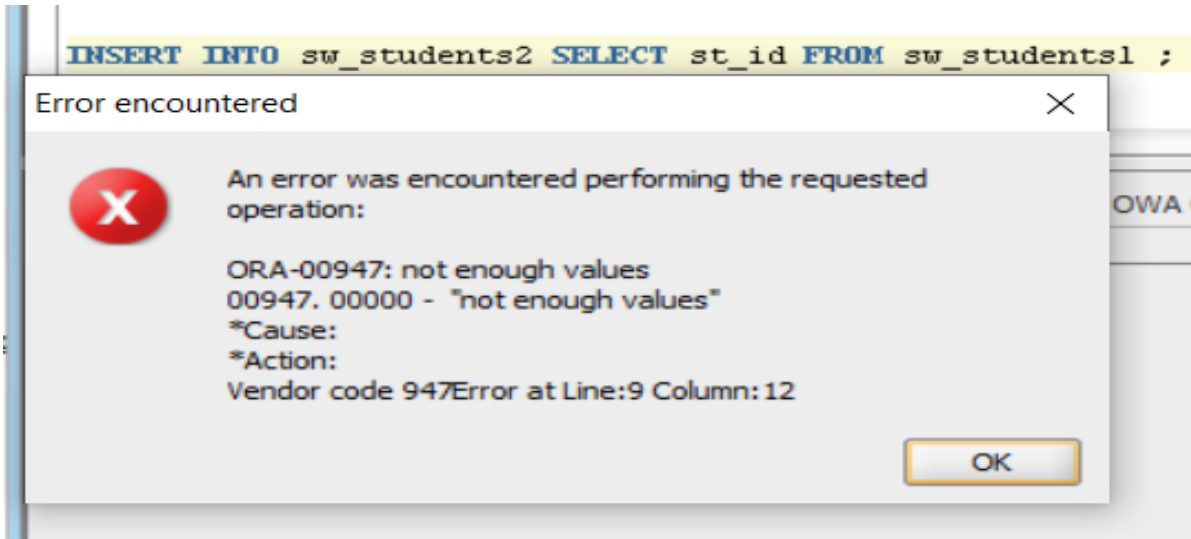
ST_ID	S_NAME	S_DOB
1	abc	18-FEB-21
2	xyz	19-FEB-21
3	hij	17-FEB-21
4	1 (null)	(null)
5	2 (null)	(null)
6	3 (null)	(null)

```
INSERT INTO sw_students2 SELECT * FROM sw_students1 ;
select * from sw_students2
```

ST_ID	S_NAME	S_DOB
1	abc	18-FEB-21
2	xyz	19-FEB-21
3	hij	17-FEB-21
4	1 (null)	(null)
5	2 (null)	(null)
6	3 (null)	(null)
7	1 abc	18-FEB-21
8	2 xyz	19-FEB-21
9	3 hij	17-FEB-21

Following are some Rules to insert data with select statement:

- 1) Column names should be in the same order & Data_Type should also be the same.
- 2) When adding a new row, you should ensure the Data_Type of the Value and the Column matches.
- 3) You should follow the Integrity Constraints, if any, defined for the Table.



INSERTING NULL INTO A COLUMN

- There are three ways to insert NULL into a column.
 1. Skip the column.
 2. Use the NULL keyword.
 3. Provide a blank space in the VALUES clause.

```
INSERT INTO sw_students (st_id, s_dob)  
VALUES (101, NULL);
```

```
INSERT INTO sw_students (st_id, s_dob)  
VALUES (101, '');
```

INSERT ALL

SYNTAX (INSERTING INTO SINGLE TABLE)

INSERT ALL

```
INSERT INTO table (column1, column2, column_n) VALUES (expr1, expr2, expr_n)
INSERT INTO table (column1, column2, column_n) VALUES (expr1, expr2, expr_n)
INSERT INTO table (column1, column2, column_n) VALUES (expr1, expr2, expr_n)
SELECT * FROM dual;
```

- In this statement, each value expression must refer to a column returned by the select list of the subquery.
- If you want to use literal values instead of the values returned by the subquery, you use the following subquery:

```
SELECT * FROM DUAL
```


SYNTAX (INSERTING INTO MULTIPLE TABLES)

INSERT ALL

INSERT INTO table1 (column1, column2, column_n) VALUES (expr1, expr2, expr_n)

INSERT INTO table1 (column1, column2, column_n) VALUES (expr1, expr2, expr_n)

INSERT INTO table2 (column1, column2, column_n) VALUES (expr1, expr2, expr_n)

SELECT * FROM dual;

```
SELECT * FROM SW_STUDENTS1
```

	ST_ID	S_NAME	S_DOB
1	1 abc		18-FEB-21
2	2 xyz		19-FEB-21
3	3 hij		17-FEB-21
4	4 OMAR		(null)
5	5 AHMED		(null)
6	6 OSMAN		(null)
7	4 AZAM		10-FEB-21

```
INSERT ALL
```

```
  INTO SW_STUDENTS1 (ST_ID,S_NAME,S_DOB) VALUES (4,'AZAM','10-FEB-2021')
```

```
  INTO EMP3 (EMPNO,ENAME,HIREDATE) VALUES (5,'AHMED','21-FEB-2010')
```

```
SELECT * FROM dual;
```

```
SELECT * FROM EMP3
```

	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
1	5 AHMED		(null)	(null)	21-FEB-10	(null)	(null)	(null)

```
SELECT * FROM SW_STUDENTS1
```

Results Script Output Explain Autotrace

Results:

ST_ID	S_NAME	S_DOB
1	abc	18-FEB-21
2	xyz	19-FEB-21
3	hij	17-FEB-21

```
INSERT ALL
  INTO SW_STUDENTS VALUES (4, 'OMAR')
  INTO SW_STUDENTS1 VALUES (5, 'AHMED')
  INTO SW_STUDENTS1 VALUES (6, 'OSMAN')
SELECT * FROM dual;
```

Error encountered



An error was encountered performing the requested operation:

ORA-00947: not enough values
00947. 00000 - "not enough values"

*Cause:

*Action:

Vendor code 947Error at Line:3 Column:7

OK

INSERT ALL

```
  INTO SW_STUDENTS1(ST_ID, S_NAME) VALUES (4, 'OMAR')
  INTO SW_STUDENTS1(ST_ID, S_NAME) VALUES (5, 'AHMED')
  INTO SW_STUDENTS1 (ST_ID, S_NAME) VALUES (6, 'OSMAN')
SELECT * FROM dual;
```

```
SELECT * FROM SW_STUDENTS1
```

Results Script Output Explain Autotrace DBMS Output OWA

Results:

ST_ID	S_NAME	S_DOB
1	abc	18-FEB-21
2	xyz	19-FEB-21
3	hij	17-FEB-21
4	OMAR	(null)
5	AHMED	(null)
6	OSMAN	(null)

INSERT ALL

```
  INTO SW_STUDENTS VALUES (4, 'OMAR', '20-FEB-2009')
  INTO SW_STUDENTS1 VALUES (5, 'AHMED', '21-FEB-2010')
SELECT * FROM dual;
```

Error encountered



An error was encountered performing the requested operation:

ORA-00913: too many values
00913. 00000 - "too many values"

*Cause:

*Action:

Vendor code 913Error at Line:2 Column:8

OK

INSERT ALL RESTRICTIONS

The Oracle multi-table insert statement is subject to the following main restrictions:

- It can be used to insert data into tables only, not views or materialized view.
- It cannot be used to insert data into remote tables.
- The number of columns in all the INSERT INTO clauses must not exceed 999.
- The subquery of the multi-table insert statement cannot use a sequence.

UPDATE COMMAND

The UPDATE Statement is used to Update/Modify existing records in a table.

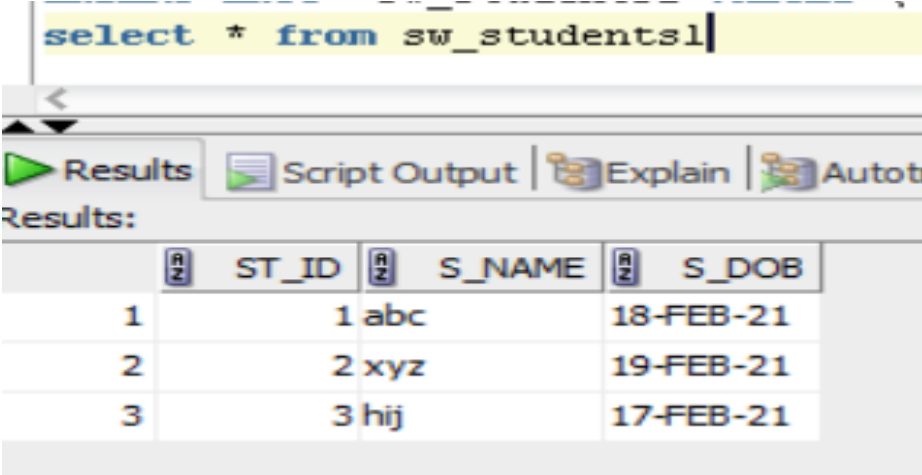
SYNTAX:

```
UPDATE table_name  
SET Column1 = Value1 [ , Column2 = Value2,...ColumnN = ValueN ]  
[ WHERE Some_Column = Some_Value ] ;
```

EXAMPLE:

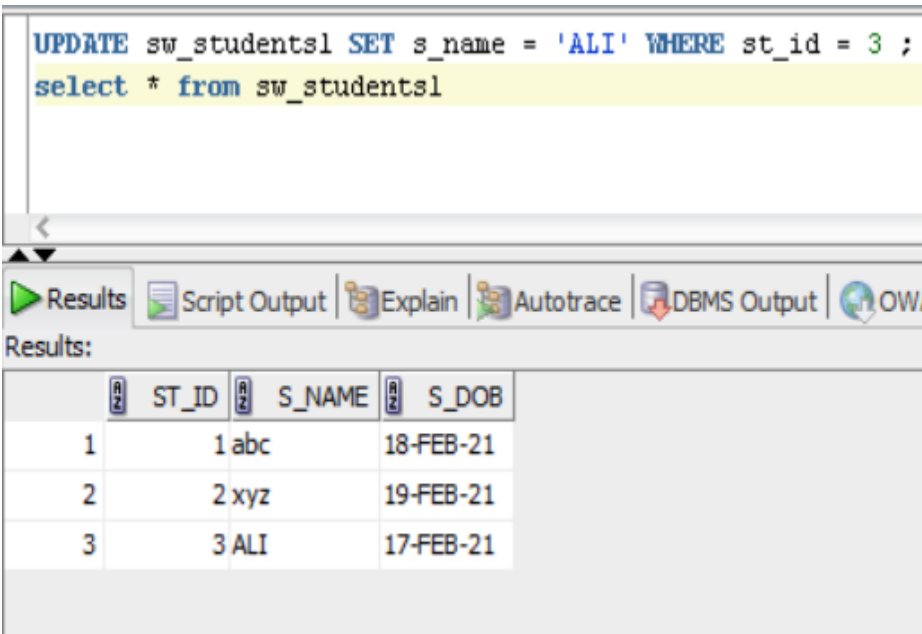
```
UPDATE sw_students  
SET s_name = 'ALI'  
WHERE s_id = 3 ;
```

NOTE: The WHERE Clause specifies which record or records that should be updated. If you Omit the WHERE clause, all records will be updated.



The screenshot shows a database query window with a SQL editor at the top containing the query: `select * from sw_students1`. Below the editor, there are tabs for 'Results', 'Script Output', 'Explain', and 'Autotrace'. The 'Results' tab is active, displaying a table with three columns: ST_ID, S_NAME, and S_DOB. The table contains three rows of data.

ST_ID	S_NAME	S_DOB
1	abc	18-FEB-21
2	xyz	19-FEB-21
3	hij	17-FEB-21



The screenshot shows a database query window with a SQL editor at the top containing two queries: `UPDATE sw_students1 SET s_name = 'ALI' WHERE st_id = 3 ;` followed by `select * from sw_students1`. Below the editor, there are tabs for 'Results', 'Script Output', 'Explain', 'Autotrace', 'DBMS Output', and 'OWA'. The 'Results' tab is active, displaying a table with three columns: ST_ID, S_NAME, and S_DOB. The table contains three rows of data, where the name of the student with ST_ID 3 has been updated to 'ALI'.

ST_ID	S_NAME	S_DOB
1	abc	18-FEB-21
2	xyz	19-FEB-21
3	ALI	17-FEB-21

```
UPDATE sw_students1 SET s_name = 'AHMED', s_name = 'OSMAN' WHERE st_id = 1 ;
```

Error encountered



An error was encountered performing the requested operation:

ORA-00957: duplicate column name
00957. 00000 - "duplicate column name"
*Cause:
*Action:
Vendor code 957Error at Line:1 Column:42

OK

OWA Output

```
UPDATE sw_students1 SET s_name = 'AHMED', s_name = 'OSMAN' WHERE st_id = 1 OR st_id = 2 ;
```

Error encountered



An error was encountered performing the requested operation:

ORA-00957: duplicate column name
00957. 00000 - "duplicate column name"
*Cause:
*Action:
Vendor code 957Error at Line:1 Column:42

OK

OWA Output

```
UPDATE sw_students1 SET s_name = 'AHMED', s_dob = '22-FEB-21' WHERE st_id = 1 OR st_id = 2;
select * from sw_students1
```

Results:

ST_ID	S_NAME	S_DOB
1	AHMED	22-FEB-21
2	AHMED	22-FEB-21
3	ALI	17-FEB-21

```
UPDATE sw_students1 SET s_name = 'ALI' WHERE st_id = 3 ;
select * from sw_students1
```

Results:

ST_ID	S_NAME	S_DOB
1	abc	18-FEB-21
2	xyz	19-FEB-21
3	ALI	17-FEB-21

The **SYNTAX** for the SQL UPDATE statement when updating a table with data from another table is:

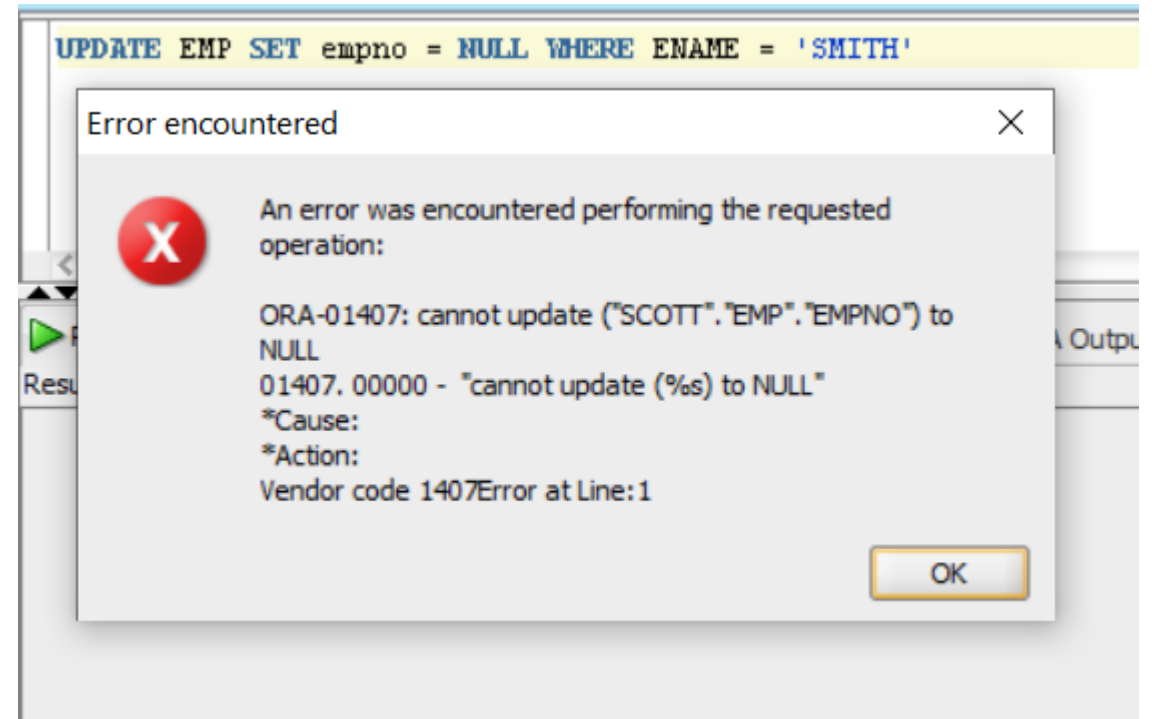
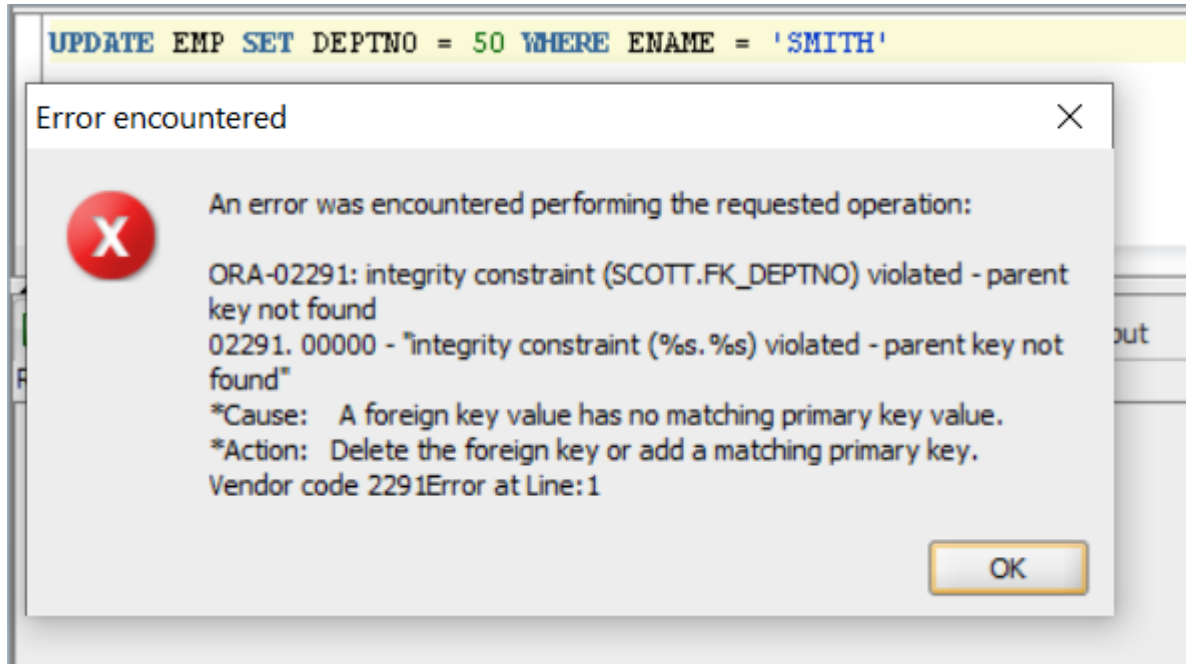
```
UPDATE table1
SET column1 = (SELECT expression1
               FROM table2
               WHERE condition)
[ WHERE condition ];
```

```
UPDATE SW_STUDENTS1 SET ST_ID = ( SELECT EMPNO FROM EMP WHERE ENAME = 'SMITH' )
WHERE S_DOB = '17-FEB-21'
select * from sw_students1
```

Results:

ST_ID	S_NAME	S_DOB
1	AHMED	22-FEB-21
2	AHMED	22-FEB-21
3	7369 ALI	17-FEB-21

UPDATE VIOLATING CONSTRAINTS



TASK A

1. Write a query that modifies emp2 table's employee 7788 's job and department number to match that of employee 7836.
2. Change the department number of all employees with the same job as of employee 7788 to the current department number of employee 7788.

DELETE COMMAND

It's used to delete rows from a Table.

SYNTAX:

DELETE FROM table_name

WHERE Some_Column = Some_Value;

- First, you specify the name of the table from which you want to delete data.
- Second, you specify which row should be deleted by using the condition in the WHERE clause. If you omit the WHERE clause, the Oracle DELETE statement removes all rows from the table.
- Note that it is faster and more efficient to use the TRUNCATE TABLE statement to delete all rows from a large table.

DELETION VIOLATING CONSTRAINTS

```
DELETE FROM DEPT WHERE DEPTNO = 10;
```

Error encountered



An error was encountered performing the requested operation:

ORA-02292: integrity constraint (SCOTT.FK_DEPTNO) violated - child record found

02292. 00000 - "integrity constraint (%s.%s) violated - child record found"

*Cause: attempted to delete a parent key value that had a foreign dependency.

*Action: delete dependencies first then parent or disable constraint.

Vendor code 2292Error at Line:1

OK

```
CREATE TABLE DEPT2 AS SELECT * FROM DEPT  
DELETE FROM DEPT2 WHERE DEPTNO = 10;  
SELECT * FROM DEPT2
```

Results Script Output Explain Autotrace DBM:

Results:

	DEPTNO	DNAME	LOC
1	20	RESEARCH	DALLAS
2	30	SALES	CHICAGO
3	40	OPERATIONS	BOSTON

TASK B

USE EMP2 AND DEPT2 TABLE TO EXECUTE THE FOLLOWING QUERIES

1. Get rid of all the employees who are working in the SALES department.
2. Apply the constraints on emp2 and dept2 which can be used to maintain Referential Integrity between the two tables.
3. After enforcement of the constraints described in query no 2, Delete the entry of department no 10 from the dept2 table.