

FSDM/CPCM-2023

Database Design and SQL

Student ID: 901142

Student Name: Roshan Shrestha

Practical Activity #1

First, to create table **STUDENTS_901142** in Oracle database, we need to execute the following query.

```
CREATE TABLE STUDENTS_901142 (  
  Student_ID VARCHAR (25) PRIMARY KEY,  
  FirstName  VARCHAR (50),  
  LastName   VARCHAR (100),  
  Telephone  VARCHAR (10),  
  Age        INTEGER,  
  City       VARCHAR (10)  
);
```

After the execution of the above query, we can use **DESC STUDENTS_901142** query to visualize the created table as below:

The screenshot shows the 'Live SQL' web interface. The SQL Worksheet contains the following queries:

```
1 CREATE TABLE STUDENTS_901142 (  
2   Student_ID VARCHAR(25) PRIMARY KEY,  
3   FirstName  VARCHAR (50),  
4   LastName   VARCHAR (100),  
5   Telephone  VARCHAR (10), Age  INTEGER,  
6   City       VARCHAR (10)  
7 );  
8  
9 DESC STUDENTS_901142  
10
```

The output shows 'Table created.' followed by the table structure:

Column	Null?	Type
STUDENT_ID	NOT NULL	VARCHAR2(25)
FIRSTNAME	—	VARCHAR2(50)
LASTNAME	—	VARCHAR2(100)
TELEPHONE	—	VARCHAR2(10)
AGE	—	NUMBER
CITY	—	VARCHAR2(10)

The above query will create table **STUDENTS_901142** and the columns are described below:

- **Student_ID:** This column stores the data type VARCHAR (25), meaning it can store up-to 25 characters.
- **FirstName:** It can store variable length string or VARCHAR data type with length constraint of maximum length of 50 characters.
- **LastName:** It can store variable length string or VARCHAR data type with length constraint of maximum length of 100 characters.
- **Telephone:** The fourth column with VARCHAR data type which can store a maximum length of 10 characters.

- **Age:** It stores the value with data type of INTEGER data type.
- **City:** It stores value of the VARCHAR data type with a maximum length of 10 characters.
- **PRIMARY KEY:** Primary key ensures each value in column is unique and non-nullable, and here it is applied to the Student_ID column.

Similarly, to create new table **COURSES_901142** and its correspondence columns, we need to execute the following query:

```
CREATE TABLE COURSES_901142 (
Student_ID VARCHAR (25),
CourseCode VARCHAR (50),
Marks INTEGER,
PRIMARY KEY (Student_ID, CourseCode),
FOREIGN KEY (Student_ID) REFERENCES STUDENTS_901142(Student_ID)
);
```

After executing the query, we can use **DESC COURSES_901142** query to preview the created table as below:

The screenshot shows a web-based SQL editor interface. At the top, there's a header with a hamburger menu, a red circle logo, and the text "Live SQL". To the right of the header are links for "Feedback", "Help", and a user profile "shrestharoshan776@gmail.com". Below the header is a toolbar with "Clear", "Find", "Actions", "Save", and a "Run" button with a play icon. The main area contains a SQL worksheet with the following code:

```
1 CREATE TABLE COURSES_901142 (
2 Student_ID VARCHAR(25),
3 CourseCode VARCHAR(50),
4 Marks INTEGER,
5 PRIMARY KEY (Student_ID, CourseCode),
6 FOREIGN KEY (Student_ID) REFERENCES STUDENTS_901142(Student_ID)
7 );
8
9 DESC COURSES_901142
```

Below the code, the result of the DESC query is displayed as a table structure for "TABLE COURSES_901142".

Column	Null?	Type
STUDENT_ID	NOT NULL	VARCHAR2(25)
COURSECODE	NOT NULL	VARCHAR2(50)
MARKS	–	NUMBER

The above query will create table **COURSES_901142** and the columns are described below:

- **Student_ID:** This column will store the student ID for the table which has datatype of VARCHAR (25) and can store up to maximum of 25 character in length.
- **CourseCode:** With maximum length of 50, it can store code of course with VARCHAR (50) datatype.
- **Marks:** It can store whole numbers as it has type INTEGER as data type.

- **PRIMARY KEY (Student_ID, CourseCode):** The Student_ID and CourseCode columns together are subject to this constraint, suggesting that each row in the database will be uniquely identified by these two columns. Every pair of values in these columns is guaranteed to be distinct and not null by the main key.
- **FOREIGN KEY (Student_ID) REFERENCES STUDENTS_901142(Student_ID):** The Student_ID column in the STUDENTS_901142 table is related to the Student_ID column in the COURSES_901142 table thanks to this constraint. It states that the values in the Student_ID column of the STUDENTS_901142 table and the Student_ID column of the COURSES_901142 table must be identical. The Student_ID values in the COURSES_901142 table must also exist as primary keys in the STUDENTS_901142 table in order to preserve referential integrity.

Practical Activity #2

1. Add a **Primary Key Constraint to Student_ID** (Table Level Constraint):
Based on the requirement from first activity, a primary key constraint has already been added to the **Student_ID** field in the **STUDENTS_901142** table. However, if we need to add a primary key, we may do so by deleting it once again. To remove the primary key constraint linked to the **Student_ID** column, we first need to determine whether any **Student_ID** dependencies need to be removed. Then, we need to run the SQL statement below.

Live SQL
 Feedback Help shrestharoshan776@gmail.com

SQL Worksheet
 Clear Find Actions Save Run

```

1 ALTER TABLE STUDENTS_901142
2 DROP PRIMARY KEY;

```

ORA-02273: this unique/primary key is referenced by some foreign keys

Because Student_ID serves as a foreign key integrity constraint in the COURSES_901142 table. We need to delete COURSES_901142 table and we will restore it using the same create statement.

≡

Live SQL

Feedback

Help

shrestharoshan776@gmail.com

SQL Worksheet

Clear

Find

Actions

Save

Run

1 DROP TABLE COURSES_901142;

Table dropped.

Now, let's try to execute the same query to alter and drop the primary key from **STUDENTS_901142** table and observe the table structure. The output is below:

≡

Live SQL

Feedback

Help

shrestharoshan776@gmail.com

SQL Worksheet

Clear

Find

Actions

Save

Run

1 ALTER TABLE STUDENTS_901142
2 DROP PRIMARY KEY;
3
4 DESC STUDENTS_901142

TABLE STUDENTS_901142

Column	Null?	Type
STUDENT_ID	–	VARCHAR2(25)
FIRSTNAME	–	VARCHAR2(50)
LASTNAME	–	VARCHAR2(100)
TELEPHONE	–	VARCHAR2(10)
AGE	–	NUMBER
CITY	–	VARCHAR2(10)

Now let's add it **Student_ID** as primary key again using below command and observe the table structure:

ALTER TABLE STUDENTS_901142
ADD CONSTRAINT STUDENTS_901142_pk PRIMARY KEY (Student_ID);

Feedback
Help
shrestharoshan776@gmail.com

SQL Worksheet
Clear
Find
Actions
Save
Run

```

1 ALTER TABLE STUDENTS_901142
2
3 ADD CONSTRAINT STUDENTS_901142_pk PRIMARY KEY (Student_ID);
4
5 DESC STUDENTS_901142
  
```

TABLE STUDENTS_901142

Column	Null?	Type
STUDENT_ID	NOT NULL	VARCHAR2(25)
FIRSTNAME	—	VARCHAR2(50)
LASTNAME	—	VARCHAR2(100)
TELEPHONE	—	VARCHAR2(10)
AGE	—	NUMBER
CITY	—	VARCHAR2(10)

As from the above table structure we can see that primary key constraint has need added to **STUDENT_ID** column, now to verify it, let's execute commands to insert same data twice and observe the output.

Feedback
Help
shrestharoshan776@gmail.com

SQL Worksheet
Clear
Find
Actions
Save
Run

```

1 INSERT INTO STUDENTS_901142(Student_ID, FirstName, LastName, Telephone, Age, City)
2 VALUES('901142', 'Roshan', 'Shrestha', '4373739305', 23, 'Pokhara')
3
4 SELECT * FROM STUDENTS_901142
5
  
```

STUDENT_ID	FIRSTNAME	LASTNAME	TELEPHONE	AGE	CITY
901142	Roshan	Shrestha	4373739305	23	Pokhara

From the above picture we can see that we were able to insert the data into the table successfully now, let's try to insert the same data to validate if the primary key constraint is working properly or not.

Feedback
Help
shrestharoshan776@gmail.com

SQL Worksheet
Clear
Find
Actions
Save
Run

```

1 INSERT INTO STUDENTS_901142(Student_ID, FirstName, LastName, Telephone, Age, City)
2 VALUES('901142','Roshan','Shrestha','4373739305', 23, 'Pokhara')
3
4
5

```

ORA-00001: unique constraint (SQL_RQRUEZSCMJSYUTGNLCNRDFBUH.STUDENTS_901142_PK) violated
ORA-06512: at "SYS.DBMS_SQL", line 1721

Finally, we can see the error message and it is verified that the **Student_ID** column has been added as primary key in **STUDENTS_901142** table.

2. Add a **NOT NULL** Constraint to **LastName** (Column Level Constraint):

To add **NOT NULL** constraint, the **LastName** column of the **STUDENTS_901142** table is modified with the **ALTER TABLE** command. The definition of the column can be modified using the **MODIFY** clause. The **LastName** column's data type, **VARCHAR (100)**, stays the same, but the **NOT NULL** constraint is added to make sure that it will never have a null value. The required query and its output are below:

ALTER TABLE STUDENTS_901142
MODIFY LastName VARCHAR (100) NOT NULL;

Feedback
Help
shrestharoshan776@gmail.com

SQL Worksheet
Clear
Find
Actions
Save
Run

```

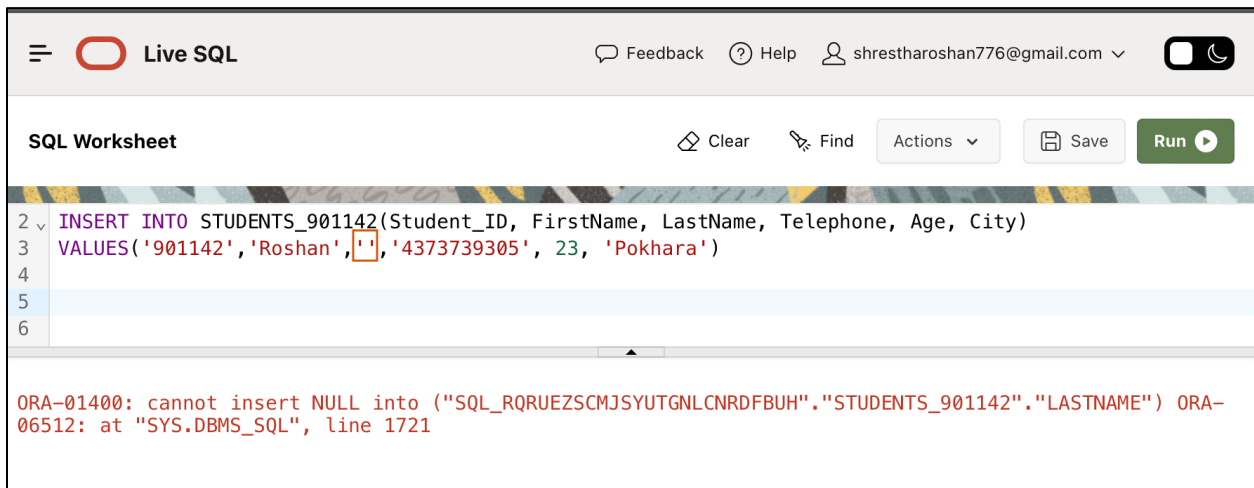
1 ALTER TABLE STUDENTS_901142
2 MODIFY LastName VARCHAR (100) NOT NULL;
3
4 DESC STUDENTS_901142
5

```

TABLE STUDENTS_901142

Column	Null?	Type
STUDENT_ID	NOT NULL	VARCHAR2(25)
FIRSTNAME	-	VARCHAR2(50)
LASTNAME	NOT NULL	VARCHAR2(100)
TELEPHONE	-	VARCHAR2(10)
AGE	-	NUMBER
CITY	-	VARCHAR2(10)

To validate if our **NOT NULL** constraint in **LastName** column we can try inserting null value and observe the error message as below:



The screenshot shows the Live SQL interface with a SQL worksheet. The query being executed is an INSERT statement into the STUDENTS_901142 table. The error message displayed is: "ORA-01400: cannot insert NULL into ("SQL_RQRUEZSCMJSYUTGNLCNRDFBUH"."STUDENTS_901142"."LASTNAME") ORA-06512: at "SYS.DBMS_SQL", line 1721".

```
2 INSERT INTO STUDENTS_901142(Student_ID, FirstName, LastName, Telephone, Age, City)
3 VALUES('901142', 'Roshan', NULL, '4373739305', 23, 'Pokhara')
4
5
6
```

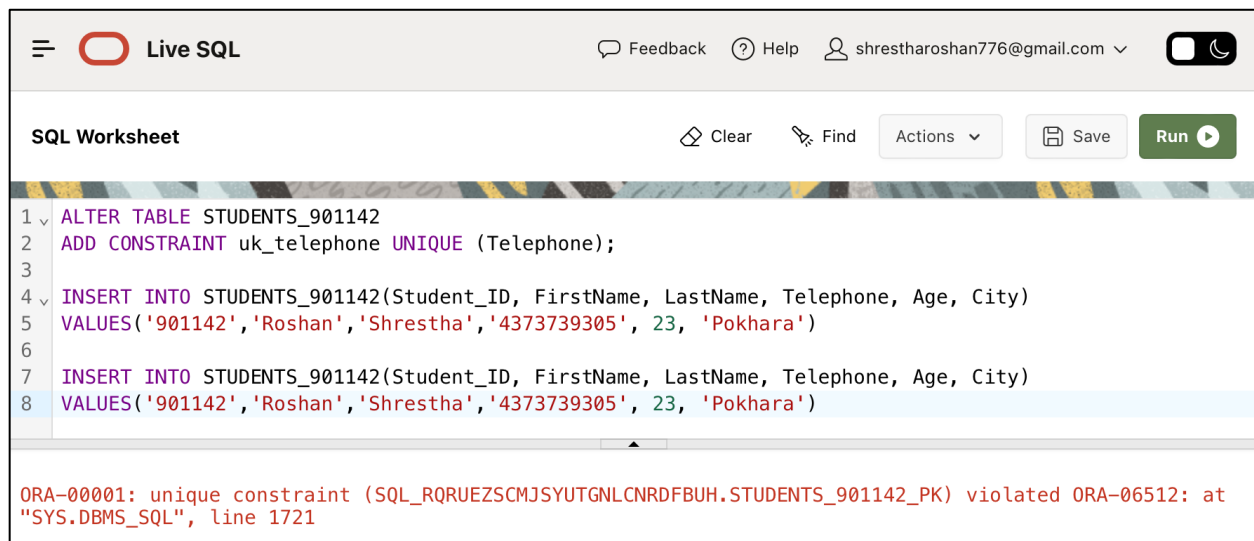
ORA-01400: cannot insert NULL into ("SQL_RQRUEZSCMJSYUTGNLCNRDFBUH"."STUDENTS_901142"."LASTNAME") ORA-06512: at "SYS.DBMS_SQL", line 1721

3. Add a **UNIQUE** Constraint to **Telephone_Number** (Table Level Constraint):

ALTER TABLE STUDENTS_901142

ADD CONSTRAINT uk_telephone UNIQUE (Telephone);

After executing the above query, the values in the Telephone column will be distinct across all table rows thanks to this **UNIQUE** constraint. It stops the table from including duplicate phone numbers. To verify if the **UNIQUE** constraint is working, we can try inserting duplicate data and observe the error message as below:



The screenshot shows the Live SQL interface with a SQL worksheet. The query being executed is an ALTER TABLE statement to add a unique constraint to the Telephone column, followed by two INSERT statements. The error message displayed is: "ORA-00001: unique constraint (SQL_RQRUEZSCMJSYUTGNLCNRDFBUH.STUDENTS_901142_PK) violated ORA-06512: at "SYS.DBMS_SQL", line 1721".

```
1 ALTER TABLE STUDENTS_901142
2 ADD CONSTRAINT uk_telephone UNIQUE (Telephone);
3
4 INSERT INTO STUDENTS_901142(Student_ID, FirstName, LastName, Telephone, Age, City)
5 VALUES('901142', 'Roshan', 'Shrestha', '4373739305', 23, 'Pokhara')
6
7 INSERT INTO STUDENTS_901142(Student_ID, FirstName, LastName, Telephone, Age, City)
8 VALUES('901142', 'Roshan', 'Shrestha', '4373739305', 23, 'Pokhara')
```

ORA-00001: unique constraint (SQL_RQRUEZSCMJSYUTGNLCNRDFBUH.STUDENTS_901142_PK) violated ORA-06512: at "SYS.DBMS_SQL", line 1721

Practical Activity #3

4. Add a **CHECK** Constraint to Age (> 18) (Column Level Constraint):

We can use the following SQL statement to add a **CHECK** constraint to the **Age** column in the **STUDENTS_901142** table, guaranteeing that the age is larger than 18.

ALTER TABLE STUDENTS_901142

ADD CHECK (Age>=18);

Now, let's try to insert the data into the table **STUDENTS_901142** to validate if the **CHECK** constraint is working as expected and observe the output as below:

The screenshot shows the Live SQL interface. The SQL Worksheet contains the following code:

```
1 ALTER TABLE STUDENTS_901142
2 ADD CHECK (Age >= 18);
3
4 INSERT INTO STUDENTS_901142(Student_ID, FirstName, LastName, Telephone, Age, City)
5 VALUES('901142', 'Roshan', 'Shrestha', '4373739305', 14, 'Pokhara')
```

The output shows "Table altered." followed by an error message: "ORA-02290: check constraint (SQL_RQRUEZSCMJSYUTGNLCNRDFBUH.SYS_C00127670714) violated ORA-06512: at 'SYS.DBMS_SQL', line 1721". The value 14 in the SQL code is highlighted with a red box.

5. Add a **DEFAULT** Constraint to City (Default city 'Toronto') (Any):
DEFAULT constraint can be applied by executing the following query:

ALTER TABLE STUDENTS_901142
MODIFY City DEFAULT 'Toronto';

The output after executing the query is below:

The screenshot shows the Live SQL interface. The SQL Worksheet contains the following code:

```
1 INSERT INTO STUDENTS_901142(Student_ID, FirstName, LastName, Telephone, Age)
2 VALUES('901142', 'Roshan', 'Shrestha', '4373739305', 23)
3
4 SELECT * FROM STUDENTS_901142
```

The output shows a table with the following data:

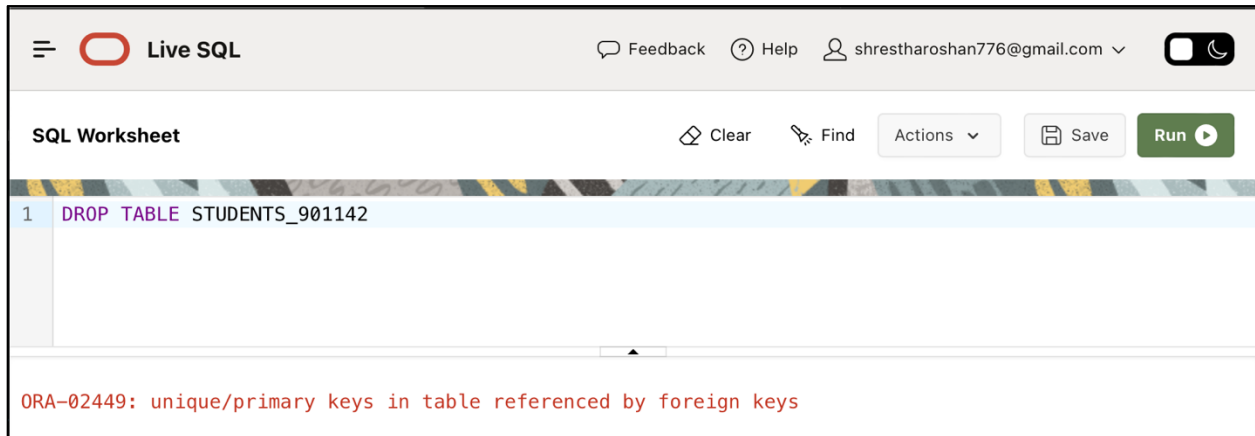
STUDENT_ID	FIRSTNAME	LASTNAME	TELEPHONE	AGE	CITY
901142	Roshan	Shrestha	4373739305	23	Toronto

The value Toronto in the table is highlighted with a red box.

Practical Activity #4

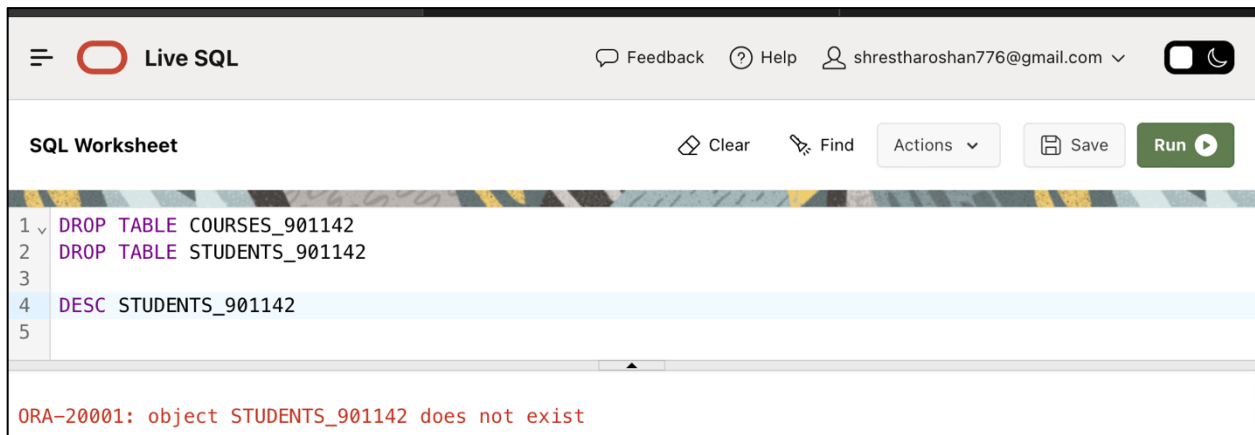
6. Create **FOREIGN KEY** Constraint - Perform **Delete / Insert** Operations to check:

The **Student_ID** and **CourseCode** columns from the **COURSES_901142** database were concatenated to form the primary key of the table, which was a requirement when constructing the **COURSES_901142** table. There is also a foreign key integrity restriction on the **Student_ID** column that refers to the **Student_ID** column in the **COURSES_901142** database. Let's try deleting the parent **STUDENTS_901142** table and observe the error message.

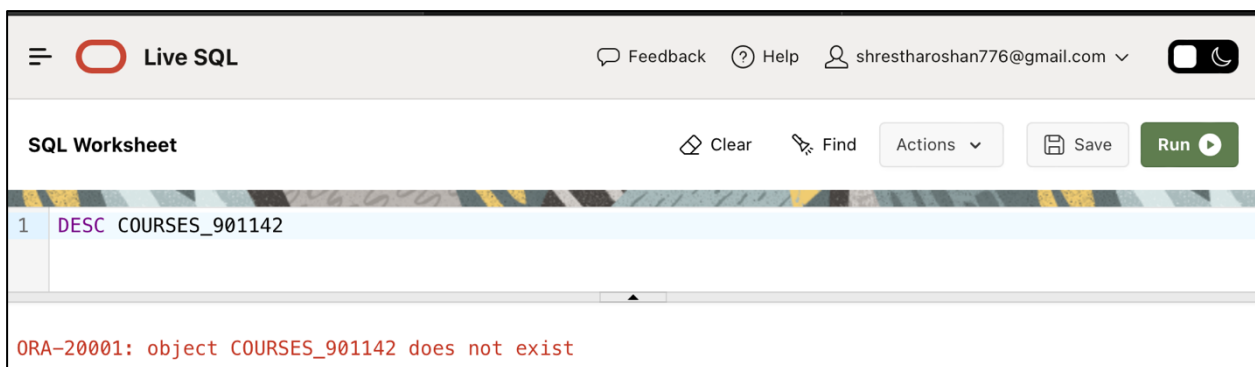


The screenshot shows the Live SQL interface with the SQL Worksheet containing the query: `DROP TABLE STUDENTS_901142`. The output area displays the error message: `ORA-02449: unique/primary keys in table referenced by foreign keys`.

Here we can see the error message that the table has primary key that is referenced by another table as foreign key. So, in order to drop the table we need to drop table **COURSES_901142** first and then drop **STUDENTS_901142**. Here is the output after executing the query:



The screenshot shows the Live SQL interface with the SQL Worksheet containing the following queries: `DROP TABLE COURSES_901142`, `DROP TABLE STUDENTS_901142`, and `DESC STUDENTS_901142`. The output area displays the error message: `ORA-20001: object STUDENTS_901142 does not exist`.



The screenshot shows the Live SQL interface with the SQL Worksheet containing the query: `DESC COURSES_901142`. The output area displays the error message: `ORA-20001: object COURSES_901142 does not exist`.

Now, to validate if table **STUDENTS_901142** and table **COURSES_901142** are linked with each other through constraints that we added previously let's try inserting data into them and observe the output.

The screenshot shows the Live SQL interface. The SQL Worksheet contains the following query:

```
1 INSERT INTO STUDENTS_901142(Student_ID, FirstName, LastName, Telephone, Age, City)
2 VALUES('901142','Roshan','Shrestha','4373739305', 23, 'Pokhara')
3
4 SELECT * FROM STUDENTS_901142
```

The query was executed successfully, and the results are displayed in a table below:

STUDENT_ID	FIRSTNAME	LASTNAME	TELEPHONE	AGE	CITY
901142	Roshan	Shrestha	4373739305	23	Pokhara

We are successful while inserting the data into **STUDENTS_901142** table, but let's try inserting data into **COURSES_901142** table with different **Student_ID** which is primary key constraint in **STUDENTS_901142** table and foreign key constraint in **COURSES_901142** table. Which is visualized in the picture below:

The screenshot shows the Live SQL interface. The SQL Worksheet contains the following query:

```
1 INSERT INTO COURSES_901142(Student_ID, CourseCode, Marks)
2 VALUES('90112', 'FSDM-101', 95)
3
4
```

The query was executed, but an error message was displayed:

```
ORA-02291: integrity constraint (SQL_HNLNDTWIJTGGTQBCWYQTUJFJ.SYS_C00127672367) violated - parent key not found
ORA-06512: at "SYS.DBMS_SQL", line 1721
```

At last let's try to insert the same data but with same **Student_ID** that we inserted into **STUDENTS_901142** table and observe the result as below:

Live SQL

Feedback

Help

shrestharoshan776@gmail.com

SQL Worksheet

Clear

Find

Actions

Save

Run

```

1 INSERT INTO COURSES_901142(Student_ID, CourseCode, Marks)
2
3 VALUES('901142', 'FSDM-101', 95)
4 SELECT * FROM COURSES_901142
5
6

```

STUDENT_ID	COURSECODE	MARKS
901142	FSDM-101	95

Lastly to validate if the deletion works, we need to follow the opposite method i.e., delete the row from **COURSES_901142** table first and then delete the row from **STUDENTS_901142**. Let's see the queries and its output as below:

Live SQL

Feedback

Help

shrestharoshan776@gmail.com

SQL Worksheet

Clear

Find

Actions

Save

Run

```

1 DELETE FROM STUDENTS_901142 where Student_ID='901142';

```

ORA-02292: integrity constraint (SQL_HNLNDTWIJTGGTQBCWYQTDUFJ.SYS_C00127672367) violated - child record found
ORA-06512: at "SYS.DBMS_SQL", line 1721

Here, as **STUDENTS_901142** table has dependency with **COURSES_901142** table, we are not allowed to delete the parent table directly as mentioned in error message. Now, let's try to delete the child table first and parent table after as shown below:

Live SQL

Feedback

Help

shrestharoshan776@gmail.com

SQL Worksheet

Clear

Find

Actions

Save

Run

```

1 DELETE FROM COURSES_901142 where Student_ID='901142';
2 DELETE FROM STUDENTS_901142 where Student_ID='901142';

```

1 row(s) deleted.

1 row(s) deleted.