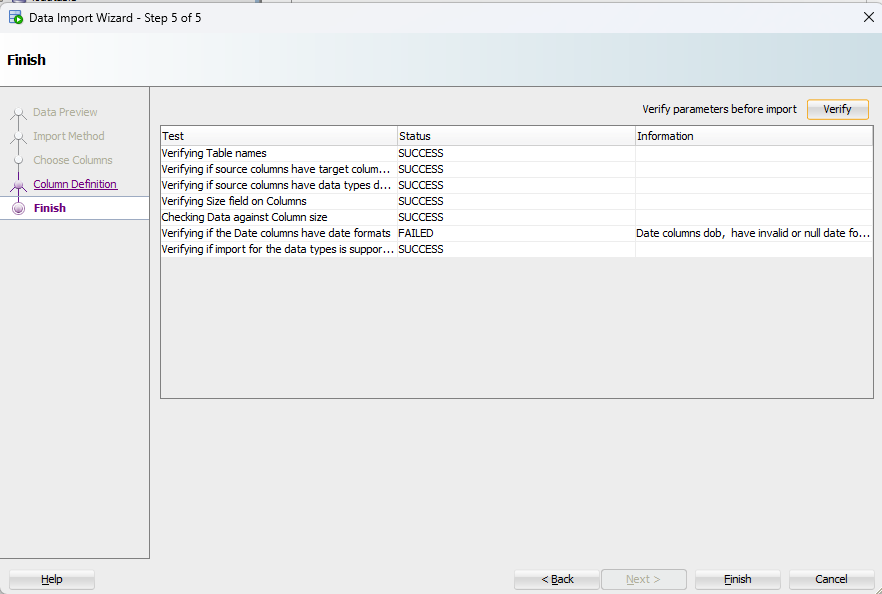
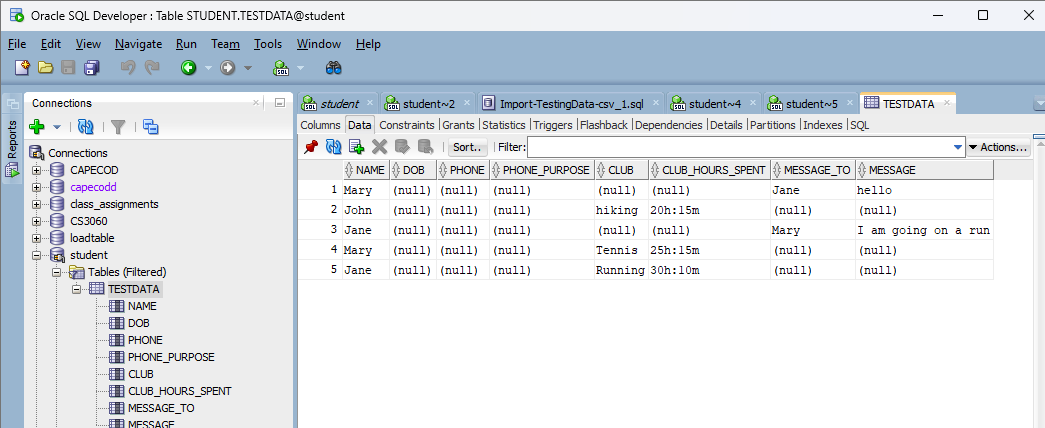
# Q1:

Load the data in the given Excel csv file to a database table as-is using SQL Developer. Show all steps and the loaded table.



tableload\_success



testdataTable

# Q2:

From the table in Q1, using SQL create the student table (student\_id, name varchar, dob) Create the keys using a student table sequence. Student name and dob cannot be NULL. Test the constraints using SQL and show the results. You have to test the primary key constraint, and the null constraint on the 2 columns.

-- create student table  
-- number, varchar, date  
-- clear table  
DROP TABLE STUDENT;  
  
-- Create student talbe  
CREATE TABLE student (  
 student\_id NUMBER PRIMARY KEY,  
 name VARCHAR2(20) NOT NULL,  
 dob DATE NOT NULL  
);  
  
-- Drop the sequence  
DROP SEQUENCE student\_id\_seq;  
  
-- create sequence for student ID   
CREATE SEQUENCE student\_id\_seq  
START WITH 1  
INCREMENT BY 1;  
  
-- load table with student information   
-- Mary   
INSERT INTO STUDENT (   
 STUDENT\_ID,  
 NAME,  
 DOB)   
  
VALUES (student\_id\_seq.nextval, 'Mary', '03-SEP-2024');  
  
-- update new student Jane   
INSERT INTO STUDENT (   
 STUDENT\_ID,  
 NAME,  
 DOB)   
  
VALUES (student\_id\_seq.nextval, 'Jane', '01-SEP-2024');  
  
select \* from student;  
  
-- test null values  
INSERT INTO STUDENT (  
 STUDENT\_ID,  
 NAME,  
 DOB)   
VALUES ('', '', '');  
  
-- Show table  
select \* from student;  
  
-- test primary key values  
INSERT INTO STUDENT (  
 STUDENT\_ID,  
 NAME,  
 DOB)   
VALUES (1, 'Michael', '30-AUG-2024');  
  
-- Show table  
select \* from student;

# Q3:

From the table in Q1 create a club table (club\_id, club\_name)

Club examples are hiking, running and tennis. Club\_id must be generated using a club sequence and the club-name cannot be null. Test the constraints using SQL and show the results. You have to test the primary key and constraint, and the null constraint on 1 column.

DROP TABLE CLUB;  
-- create club table  
CREATE TABLE CLUB (  
 club\_id NUMBER PRIMARY KEY,  
 club\_name VARCHAR2(20) not null  
);  
  
-- drop sequence  
drop sequence club\_id\_seq;  
  
-- generate sequence for club ID  
CREATE SEQUENCE club\_id\_seq  
START WITH 1  
INCREMENT BY 1;  
  
-- insert new rows for hiking, running and tennis  
INSERT INTO CLUB (club\_id, club\_name)  
VALUES (club\_id\_seq.nextval, 'HIKING');  
  
INSERT INTO CLUB (club\_id, club\_name)  
VALUES (club\_id\_seq.nextval, 'RUNNING');  
  
INSERT INTO CLUB (club\_id, club\_name)  
VALUES (club\_id\_seq.nextval, 'TENNIS');  
  
-- test primary key and "NULL" values  
INSERT INTO CLUB (club\_id, club\_name)  
VALUES (1,'HIKING');  
  
-- check output  
select \* from club;  
insert into club (club\_id, club\_name) values (5, '');  
  
-- check output  
select \* from club;

# Q4:

From the table in Q1 create a messages table (message\_from, message\_to, message) Example message is from Mary to Jane and Hello. message\_from & message\_to should be foreign keys to the student table, and the message cannot be null. message\_from & message\_to are student\_ids. Test the constraints using SQL and show the results. You have to test the foreign key and constraint, and the null constraint on 1 column.

-- Q4  
-- create messages table  
drop table messages;  
  
-- messages\_from and message\_to foreign keys in the student table  
create table messages (   
 message\_id number primary key,  
 message\_from number not null,  
 message\_to number not null,  
 message varchar2(50) not null, -- messages cannot be 'NULL'  
  
-- message from and to are student\_ids  
foreign key (message\_from) references student(student\_id),   
foreign key (message\_to) references student(student\_id));  
  
-- Create a sequence for message\_id  
create sequence message\_id\_seq  
start with 1  
increment by 1;  
  
-- Test message Mary to Jane is "hello"  
insert into messages (message\_id, message\_from, message\_to, message)  
values (message\_id\_seq.nextval,   
 (select student\_id from STUDENT where name = 'Mary'),   
 (select student\_id from STUDENT where name = 'Jane'),  
 'Hello');  
  
-- show output   
select \* from messages;  
  
-- Test foreign key constraint   
-- (non-existent student\_id 999 for message\_from)  
insert into messages (message\_id, message\_from, message\_to, message)  
values (message\_id\_seq.nextval,   
 (select student\_id from STUDENT where student\_id = 999),   
 (select student\_id from STUDENT where student\_id = 1),  
 'Hello');  
  
-- Test NULL constraint  
insert into messages (message\_id, message\_from, message\_to, message)  
values (message\_id\_seq.nextval,   
 (select student\_id from STUDENT where name = 'Mary'),   
 (select student\_id from STUDENT where name = 'Jane'),  
 NULL);  
  
-- show output  
select \* from messages;  
  
-- create a table called club\_time\_spent   
drop table phone;  
  
-- cols are student\_id and time\_in\_minutes  
create table phone (  
  
student\_id number primary key,  
time number,   
  
foreign key (student\_id) references student(student\_id));

You will also need to create a table called Club\_Time\_Spent (student\_id, time\_in\_minutes). This table also should have the foreign key constraints and the null constraints. You will need to do data wrangling to store the time in minutes. You will use this table in an SQL later.

-- Q4 cont.  
-- create a table called club\_time\_spent   
drop table club\_time\_spent;  
  
-- cols are student\_id and time\_in\_minutes  
create table club\_time\_spent(   
 time\_in\_minutes time primary key,  
 student\_id number,  
foreign key (student\_id) references student(student\_id));

# Q5:

From the table in Q1 create the phone table (student\_id, phone, phone\_purpose) Phone\_purpose can only be cell, work or home. Test the constraints using SQL and show the results. You have to test the foreign key constraint, and the constraint on the phone\_purpose column.

# Q6:

For the Student table create a history table that stores the old student row with timestamp (Sysdate in Oracle) on update of student row, using a PL/SQL procedure. Test this by updating a student row and thus creating an entry in the student-history table. Show the before and after of the tables.

# Q7:

Create a View that shows the message-from (student), the message-to (student), their dobs, their phone and the message sent, and order by dob of message-from student. dob is a date column and not a string. Do a select from the View to show all the rows. dob should show as MM-DD-YYYY and phone-number should show in the format XXX-XXX-XXXX.

# Q8:

Using a Java Metadata program, show the metadata for the Student and Club tables only. There is no need to show the DB metadata, just the information for the 2 tables.

# Q9:

Using a Java program SQL inject the Student table and using a Java Prepared statement show that the SQL injection can be prevented. Show your work by running the prograqm and output.

# Q10.

Write an SQL query that shows all students, their dob, their cell phone numbers, the clubs they are members of and the total time they spent in the club in minutes. Test it in SQL developer, and then run it in Java and show the ResultSet metadata.