

Q1 Create MY_EMPLOYEE table having attributes employee_id, first_name, last_name, userid, salary. Choose appropriate data types for each field. Make employee_id as primary key.

Query

```
CREATE TABLE my_employee (  
    employee_id VARCHAR(25) PRIMARY KEY,  
    first_name VARCHAR(25),  
    last_name VARCHAR(25),  
    user_id VARCHAR(25),  
    salary NUMBER(7, 2)  
);
```

Output

The screenshot displays a web-based SQL editor interface. On the left, a 'Navigator' pane shows a tree view of database objects under 'My Schema'. The 'MY_EMPLOYEE' table is selected, and its columns (EMPLOYEE_ID, FIRST_NAME, LAST_NAME, USER_ID, SALARY) are listed below it. The main editor area, titled '[SQL Worksheet]*', contains the SQL code for creating the table. Below the editor, a 'Script output' tab is active, showing the execution results. The output includes the SQL command, a truncated version of the table definition, a confirmation message 'Table MY_EMPLOYEE created.', and the elapsed time '00:00:00.015'.

```
1 CREATE TABLE my_employee (  
2     employee_id VARCHAR(25) PRIMARY KEY,  
3     first_name VARCHAR(25),  
4     last_name VARCHAR(25),  
5     user_id VARCHAR(25),  
6     salary NUMBER(7, 2)  
7 );  
8
```

Query result **Script output** DBMS output Explain Plan SQL hist

SQL> CREATE TABLE my_employee (
 employee_id VARCHAR(25) PRIMARY KEY,
 first_name VARCHAR(25),
 last_name VARCHAR(25),...
Show more...

Table MY_EMPLOYEE created.

Elapsed: 00:00:00.015

Q2

Rename MY_EMPLOYEE table as MY_STAFF.

Query

```
RENAME my_employee TO my_staff;
```

Output

The screenshot displays the Live SQL interface. On the left, the Navigator pane shows a tree structure under 'My Schema' with 'Tables' expanded. The 'MY_STAFF' table is selected, showing its columns: EMPLOYEE_ID, FIRST_NAME, LAST_NAME, USER_ID, and SALARY. The main editor on the right contains the following SQL code:

```
1 CREATE TABLE my_employee (  
2     employee_id VARCHAR(25) PRIMARY KEY,  
3     first_name VARCHAR(25),  
4     last_name VARCHAR(25),  
5     user_id VARCHAR(25),  
6     salary NUMBER(7, 2)  
7 );  
8  
9 RENAME my_employee TO my_staff;  
10
```

Below the editor, the 'Script output' tab is active, showing the execution results:

```
SQL> CREATE TABLE my_employee (  
    employee_id VARCHAR(25) PRIMARY KEY,  
    first_name VARCHAR(25),  
    last_name VARCHAR(25),...  
Show more...  
  
Table MY_EMPLOYEE created.  
Elapsed: 00:00:00.015  
  
SQL> RENAME my_employee TO my_staff  
  
Table renamed.  
Elapsed: 00:00:00.009
```

Q3

Describe the structure of the MY_STAFF table to identify the column names.

Query

```
describe my_staff;
```

Output

The screenshot shows a Live SQL interface with a worksheet containing the following SQL statements:

```

1 CREATE TABLE my_employee (
2     employee_id VARCHAR(25) PRIMARY KEY,
3     first_name VARCHAR(25),
4     last_name VARCHAR(25),
5     user_id VARCHAR(25),
6     salary NUMBER(7, 2)
7 );
8
9 RENAME my_employee TO my_staff;
10
11 describe my_staff;

```

The output pane shows the results of the executed queries:

SQL> RENAME my_employee TO my_staff

Table renamed.

Elapsed: 00:00:00.009

SQL> describe my_staff

Name	Null?	Type
EMPLOYEE_ID	NOT NULL	VARCHAR2(25)
FIRST_NAME		VARCHAR2(25)
LAST_NAME		VARCHAR2(25)
USER_ID		VARCHAR2(25)
SALARY		NUMBER(7,2)

Q4

Add the first row of data to the MY_STAFF table from the following sample data. Do not list

the columns in the INSERT clause.

Query

```
INSERT INTO my_staff
VALUES ('1', 'Ralph', 'Patel', 'rpatel', 895);
```

Output

The screenshot shows the Live SQL interface. On the left is a Navigator panel with a tree view of the database schema. The 'My Schema' is expanded, showing a table named 'MY_STAFF'. The columns listed are EMPLOYEE_ID, FIRST_NAME, LAST_NAME, USER_ID, and SALARY. The main area on the right is the SQL Worksheet, which contains the following SQL code:

```
1 CREATE TABLE my_employee (
2     employee_id VARCHAR(25) PRIMARY KEY,
3     first_name VARCHAR(25),
4     last_name VARCHAR(25),
5     user_id VARCHAR(25),
6     salary NUMBER(7, 2)
7 );
8
9 RENAME my_employee TO my_staff;
10
11 describe my_staff;
12
13 INSERT INTO my_staff
14 VALUES ('1', 'Ralph', 'Patel', 'rpatel', 895);
15
```

Below the SQL code, the 'Script output' tab is selected, showing the results of the executed queries. The first query is 'describe my_staff', which returns the following table structure:

Name	Null?	Type
EMPLOYEE_ID	NOT NULL	VARCHAR2(25)
FIRST_NAME		VARCHAR2(25)
LAST_NAME		VARCHAR2(25)
USER_ID		VARCHAR2(25)
SALARY		NUMBER(7,2)

The second query is 'INSERT INTO my_staff VALUES ('1', 'Ralph', 'Patel', 'rpatel', 895);', which returns the message '1 row inserted.' and an elapsed time of 00:00:00.017.

Q5

Populate the MY_STAFF table with the second row of sample data from the preceding list.

This time, list the columns explicitly in the INSERT clause.

Query

```
INSERT INTO my_staff (employee_id, first_name, last_name, user_id, salary)
VALUES ('3', 'Ben', 'Biri', 'bbiri', 1100);
```

```
INSERT INTO my_staff (employee_id, first_name, last_name, user_id, salary)
VALUES ('4', 'Chad', 'Newman', 'cnewman', 750);
```

```
INSERT INTO my_staff (employee_id, first_name, last_name, user_id, salary)
VALUES ('5', 'Audrey', 'Ropeburn', 'aropebur', 1550);
```

Output

The screenshot shows a SQL IDE interface with a 'Live SQL' tab. The left sidebar displays a 'Navigator' with a tree view of the database schema. Under 'My Schema', the 'Tables' section is expanded, showing 'MY_STAFF' selected. The main editor area shows a SQL script with the following content:

```

6      salary NUMBER(7, 2)
7  );
8
9  RENAME my_employee TO my_staff;
10
11 describe my_staff;
12
13 INSERT INTO my_staff
14 VALUES ('1', 'Ralph', 'Patel', 'rpatel', 895);
15
16 INSERT INTO my_staff (employee_id, first_name, last_name, user_id, salary)
17 VALUES ('3', 'Ben', 'Biri', 'bbiri', 1100);
18
19 INSERT INTO my_staff (employee_id, first_name, last_name, user_id, salary)
20 VALUES ('4', 'Chad', 'Newman', 'cnewman', 750);
21
22 INSERT INTO my_staff (employee_id, first_name, last_name, user_id, salary)
23 VALUES ('5', 'Audrey', 'Ropeburn', 'aropebur', 1550);
24

```

Below the script, the 'Script output' tab is active, showing the execution results for two queries:

```

SQL> INSERT INTO my_staff (employee_id, first_name, last_name, user_id, salary)
VALUES ('4', 'Chad', 'Newman', 'cnewman', 750)

1 row inserted.
Elapsed: 00:00:00.001

SQL> INSERT INTO my_staff (employee_id, first_name, last_name, user_id, salary)
VALUES ('5', 'Audrey', 'Ropeburn', 'aropebur', 1550)

1 row inserted.
Elapsed: 00:00:00.001

```

Q6

Confirm your addition to the table.

Query

```
SELECT * FROM my_staff;
```

Output

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	USER_ID	SALARY
1	1	Ralph	Patel	rpatel	895
2	3	Ben	Biri	bbiri	1100
3	4	Chad	Newman	cnewman	750
4	5	Audrey	Ropeburn	aropebur	1550

Question 7

Populate the table with the next two rows of sample data by running the INSERT statement in such a way that you concatenate the first letter of the first name and the first seven characters of the last name to produce the user ID, instead of hardcoding it.

Query

```
INSERT INTO my_staff (employee_id, first_name, last_name, user_id, salary)
VALUES ('6', 'Susan', 'Jones', SUBSTR('Susan', 1, 1) || SUBSTR('Jones', 1, 7), 1200);

INSERT INTO my_staff (employee_id, first_name, last_name, user_id, salary)
VALUES ('7', 'Mike', 'McMannus', SUBSTR('Mike', 1, 1) || SUBSTR('McMannus', 1, 7),
1300);
```

Output

The screenshot shows the Live SQL interface with a SQL worksheet. The left sidebar displays a database schema with a table named `MY_STAFF` containing columns: `EMPLOYEE_ID`, `FIRST_NAME`, `LAST_NAME`, `USER_ID`, and `SALARY`. The main area shows a SQL script with several `INSERT` statements. The bottom panel shows the execution results for two of these statements.

```

14 VALUES ('1', 'Ralph', 'Patel', 'rpatel', 895);
15
16 INSERT INTO my_staff (employee_id, first_name, last_name, user_id, salary)
17 VALUES ('3', 'Ben', 'Biri', 'bbiri', 1100);
18
19 INSERT INTO my_staff (employee_id, first_name, last_name, user_id, salary)
20 VALUES ('4', 'Chad', 'Newman', 'cnewman', 750);
21
22 INSERT INTO my_staff (employee_id, first_name, last_name, user_id, salary)
23 VALUES ('5', 'Audrey', 'Ropeburn', 'aropebur', 1550);
24
25 SELECT * FROM my_staff;
26
27 INSERT INTO my_staff (employee_id, first_name, last_name, user_id, salary)
28 VALUES ('6', 'Susan', 'Jones', SUBSTR('Susan', 1, 1) || SUBSTR('Jones', 1, 7), 1200);
29
30 INSERT INTO my_staff (employee_id, first_name, last_name, user_id, salary)
31 VALUES ('7', 'Mike', 'McMannus', SUBSTR('Mike', 1, 1) || SUBSTR('McMannus', 1, 7), 1300);
32

```

Query result | **Script output** | DBMS output | Explain Plan | SQL history

SQL> INSERT INTO my_staff (employee_id, first_name, last_name, user_id, salary)
VALUES ('6', 'Susan', 'Jones', SUBSTR('Susan', 1, 1) || SUBSTR('Jones', 1, 7), 1200)

1 row inserted.
Elapsed: 00:00:00.002

SQL> INSERT INTO my_staff (employee_id, first_name, last_name, user_id, salary)
VALUES ('7', 'Mike', 'McMannus', SUBSTR('Mike', 1, 1) || SUBSTR('McMannus', 1, 7), 1300)

1 row inserted.
Elapsed: 00:00:00.001

Question 8

Confirm your additions to the table and make the data additions permanent.

Query

```
select * from my_staff;
commit;
```

Output

The screenshot shows a 'Live SQL' interface with a 'Navigator' on the left and a main editor on the right. The 'Navigator' shows a schema named 'My Schema' with a table 'MY_STAFF' selected. The main editor contains a SQL script with the following content:

```

17 VALUES ('3', 'Ben', 'Biri', 'bbiri', 1100);
18
19 INSERT INTO my_staff (employee_id, first_name, last_name, user_id, salary)
20 VALUES ('4', 'Chad', 'Newman', 'cnewman', 750);
21
22 INSERT INTO my_staff (employee_id, first_name, last_name, user_id, salary)
23 VALUES ('5', 'Audrey', 'Ropeburn', 'aropebur', 1550);
24
25 SELECT * FROM my_staff;
26
27 INSERT INTO my_staff (employee_id, first_name, last_name, user_id, salary)
28 VALUES ('6', 'Susan', 'Jones', SUBSTR('Susan', 1, 1) || SUBSTR('Jones', 1, 7), 1200);
29
30 INSERT INTO my_staff (employee_id, first_name, last_name, user_id, salary)
31 VALUES ('7', 'Mike', 'McHannus', SUBSTR('Mike', 1, 1) || SUBSTR('McHannus', 1, 7), 1300);
32
33 select * from my_staff;
34
35 commit;
36

```

Below the script, the 'Script output' tab is active, showing the execution results:

```

SQL> INSERT INTO my_staff (employee_id, first_name, last_name, user_id, salary)
VALUES ('7', 'Mike', 'McHannus', SUBSTR('Mike', 1, 1) || SUBSTR('McHannus', 1, 7), 1300)

1 row inserted.
Elapsed: 00:00:00.001

SQL> commit

Commit complete.
Elapsed: 00:00:00.000

```

Q9

Change the last name of employee 3 to Drexler.

Query

```

UPDATE my_staff
SET last_name = 'Drexler'
WHERE employee_id = '3';

```

Output

The screenshot shows the Live SQL interface with a SQL worksheet containing the following code:

```

26
27 INSERT INTO my_staff (employee_id, first_name, last_name, user_id, salary)
28 VALUES ('6', 'Susan', 'Jones', SUBSTR('Susan', 1, 1) || SUBSTR('Jones', 1, 7), 1200);
29
30 INSERT INTO my_staff (employee_id, first_name, last_name, user_id, salary)
31 VALUES ('7', 'Mike', 'McMannus', SUBSTR('Mike', 1, 1) || SUBSTR('McMannus', 1, 7), 1300);
32
33 select * from my_staff;
34
35 commit;
36
37 UPDATE my_staff
38 SET last_name = 'Drexler'
39 WHERE employee_id = '3';
40
41 commit;
42 SELECT * FROM my_staff
43 WHERE employee_id = '3';
44
45

```

The Query result section shows the following table:

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	USER_ID	SALARY
1	3	Ben	Drexler	bbiri	1100

Q10

Change the salary to 1000 for all employees with a salary less than 900. Verify your changes to the table.

Query

```

UPDATE my_staff
SET salary = 1000
WHERE salary < 900;

SELECT * FROM my_staff
ORDER BY employee_id;

```

Output

Live SQL Worksheet Library

Navigator Files

My Schema

Tables

Search objects

CATEGORY
 MYCUSTOMER
 MY_STAFF
 EMPLOYEE_ID
 FIRST_NAME
 LAST_NAME
 USER_ID
 SALARY
 NEWCUSTOMER
 ORDERS
 PRODUCT

```

33 select * from my_staff;
34
35 commit;
36
37 UPDATE my_staff
38 SET last_name = 'Drexler'
39 WHERE employee_id = '3';
40
41 commit;
42 SELECT * FROM my_staff
43 WHERE employee_id = '3';
44
45 UPDATE my_staff
46 SET salary = 1000
47 WHERE salary < 900;
48
49 SELECT * FROM my_staff
50 ORDER BY employee_id;
51
52
  
```

Query result Script output DBMS output Explain Plan SQL history

Download Execution time: 0.004 seconds

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	USER_ID	SALARY
1	1	Ralph	Patel	rpatel	1000
2	3	Ben	Drexler	bbiri	1100
3	4	Chad	Newman	cnewman	1000
4	5	Audrey	Ropeburn	aropebur	1550
5	6	Susan	Jones	SJones	1200
6	7	Mike	McMannus	MMcMannu	1300

Q11

Delete Betty Dancs from the MY_STAFF table.

Query

```

DELETE FROM my_staff
WHERE first_name = 'Betty' AND last_name = 'Dancs';
  
```

Output

The screenshot shows the Live SQL interface with a sidebar on the left and a main workspace on the right.

Navigator:

- My Schema
 - Tables
 - MY_STAFF (selected)
 - EMPLOYEE_ID
 - FIRST_NAME
 - LAST_NAME
 - USER_ID
 - SALARY
 - NEWCUSTOMER
 - ORDERS
 - PRODUCT

SQL Worksheet [SQL Worksheet]*

```

36
37 UPDATE my_staff
38 SET last_name = 'Drexler'
39 WHERE employee_id = '3';
40
41 commit;
42 SELECT * FROM my_staff
43 WHERE employee_id = '3';
44
45 UPDATE my_staff
46 SET salary = 1000
47 WHERE salary < 900;
48
49 SELECT * FROM my_staff
50 ORDER BY employee_id;
51
52 DELETE FROM my_staff
53 WHERE first_name = 'Betty' AND last_name = 'Dancs';
54
55

```

Script output

```

SQL> UPDATE my_staff
      SET salary = 1000
      WHERE salary < 900

0 rows updated.

Elapsed: 00:00:00.005

SQL> DELETE FROM my_staff
      WHERE first_name = 'Betty' AND last_name = 'Dancs'

0 rows deleted.

Elapsed: 00:00:00.007

```

Q12

Confirm your changes to the table and Commit all pending changes.

Query

```

SELECT * FROM my_staff
ORDER BY employee_id;
COMMIT;

```

Output

The screenshot shows the Live SQL interface. On the left, the Navigator pane shows the database schema with 'My Schema' selected. Under 'Tables', 'MY_STAFF' is expanded, showing columns: EMPLOYEE_ID, FIRST_NAME, LAST_NAME, USER_ID, and SALARY. The main area displays a SQL script in a worksheet titled '[SQL Worksheet]*'. The script includes a COMMIT, a SELECT statement, an UPDATE statement, another SELECT statement, a DELETE statement, and a final SELECT statement. Below the script, the 'Query result' tab is active, showing the execution time as 0.006 seconds and a table of results.

```

40
41 COMMIT;
42 SELECT * FROM my_staff
43 WHERE employee_id = '3';
44
45 UPDATE my_staff
46 SET salary = 1000
47 WHERE salary < 900;
48
49 SELECT * FROM my_staff
50 ORDER BY employee_id;
51
52 DELETE FROM my_staff
53 WHERE first_name = 'Betty' AND last_name = 'Dancs';
54
55 SELECT * FROM my_staff
56 ORDER BY employee_id;
57 COMMIT;
58
59

```

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	USER_ID	SALARY
1	1	Ralph	Patel	rpatel	1000
2	3	Ben	Drexler	bbiri	1100
3	4	Chad	Newman	cnewman	1000
4	5	Audrey	Ropeburn	aropebur	1550
5	6	Susan	Jones	SJones	1200
6	7	Mike	McMannus	MMcMannu	1300

Q13

Populate the table with the last row of sample data by modifying the insert statement that you created in step 7. Confirm your addition to the table. Create savepoint.

Query

```

INSERT INTO my_staff (employee_id, first_name, last_name, user_id, salary)
VALUES ('8', 'Jennifer', 'Whalen', SUBSTR('Jennifer', 1, 1) || SUBSTR('Whalen', 1, 7),
1450);
SELECT * FROM my_staff
WHERE employee_id = '8';
SAVEPOINT after_jennifer_insert;

```

Output

The screenshot shows a Live SQL interface with a SQL worksheet. The script in the worksheet is as follows:

```

46 SET salary = 1000
47 WHERE salary < 900;
48
49 SELECT * FROM my_staff
50 ORDER BY employee_id;
51
52 DELETE FROM my_staff
53 WHERE first_name = 'Betty' AND last_name = 'Dancs';
54
55 SELECT * FROM my_staff
56 ORDER BY employee_id;
57 COMMIT;
58
59 INSERT INTO my_staff (employee_id, first_name, last_name, user_id, salary)
60 VALUES ('8', 'Jennifer', 'Whalen', SUBSTR('Jennifer', 1, 1) || SUBSTR('Whalen', 1, 7), 1450);
61 SELECT * FROM my_staff
62 WHERE employee_id = '8';
63 SAVEPOINT after_jennifer_insert;
64
65

```

The query results are displayed in a table below the script:

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	USER_ID	SALARY
1	8	Jennifer	Whalen	JWhalen	1450

Q14

Empty the entire table and confirm that the table is empty.

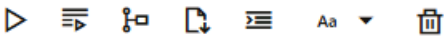
Query

```

DELETE FROM my_staff;
SELECT * FROM my_staff;
COMMIT;

```

Output



[SQL Worksheet]* 

```

50 ORDER BY employee_id;
51
52 DELETE FROM my_staff
53 WHERE first_name = 'Betty' AND last_name = 'Dancs';
54
55 SELECT * FROM my_staff
56 ORDER BY employee_id;
57 COMMIT;
58
59 INSERT INTO my_staff (employee_id, first_name, last_name, user_id, salary)
60 VALUES ('8', 'Jennifer', 'Whalen', SUBSTR('Jennifer', 1, 1) || SUBSTR('Whalen', 1, 7), 1450);
61 SELECT * FROM my_staff
62 WHERE employee_id = '8';
63 SAVEPOINT after_jennifer_insert;
64
65 DELETE FROM my_staff;
66 SELECT * FROM my_staff;
67 COMMIT;
68

```

Query result Script output DBMS output Explain Plan SQL history

  Download Execution time: 0.004 seconds

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	USER_ID	SALARY
No items to display.				

Q15

Discard the most recent DELETE operation without discarding the earlier INSERT operation.

Confirm that the new row is still intact. Make the data addition now permanent.

Query

```

ROLLBACK TO after_jennifer_insert;
SELECT * FROM my_staff
WHERE employee_id = '8';
COMMIT;

```

Output

Q16

Create MY_DEPART table with necessary attributes. Choose appropriate data types for each

field. Make depart_id as primary key.

Query

```
CREATE TABLE my_depart (  
    depart_id VARCHAR2(10) PRIMARY KEY,  
    depart_name VARCHAR2(50)  
);
```

Output

The screenshot shows an SQL worksheet interface with a toolbar at the top containing icons for running, saving, and other actions. The main area displays the following SQL query:

```
1 CREATE TABLE my_depart (  
2     depart_id VARCHAR2(10) PRIMARY KEY,  
3     depart_name VARCHAR2(50)  
4 );  
5
```

Below the query editor, there are tabs for "Query result", "Script output", "DBMS output", "Explain Plan", and "SQL history". The "Script output" tab is currently selected, showing the following output:

```
SQL> COMMIT  
  
Commit complete.  
Elapsed: 00:00:00.000  
  
SQL> CREATE TABLE my_depart (  
        depart_id VARCHAR2(10) PRIMARY KEY,  
        depart_name VARCHAR2(50)  
    )  
  
Table MY_DEPART created.  
Elapsed: 00:00:00.021
```

Q17

Modify MY_STAFF table to include depart_id as foreign key. What happens to previous 5

rows with this change in the MY_STAFF table?

Query

```
ALTER TABLE my_staff  
ADD depart_id VARCHAR2(10);  
  
ALTER TABLE my_staff  
ADD CONSTRAINT fk_depart  
FOREIGN KEY (depart_id)  
REFERENCES my_depart(depart_id);
```

Output

Live SQL

Worksheet

Library

Navigator

Files

My Schema

Tables

Search objects

CATEGORY

MYCUSTOMER

MY_DEPART

MY_STAFF

NEWCUSTOMER

ORDERS

PRODUCT

[SQL Worksheet]*

1 CREATE TABLE my_depart (
2 | depart_id VARCHAR2(10) PRIMARY KEY,
3 | depart_name VARCHAR2(50)
4 |);
5
6 ALTER TABLE my_staff
7 ADD depart_id VARCHAR2(10);
8
9 ALTER TABLE my_staff
10 ADD CONSTRAINT fk_depart
11 FOREIGN KEY (depart_id)
12 REFERENCES my_depart(depart_id);
13
14

Query result

Script output

DBMS output

Explain Plan

SQL histo

SQL> ALTER TABLE my_staff
ADD depart_id VARCHAR2(10)

Table MY_STAFF altered.
Elapsed: 00:00:00.026

SQL> ALTER TABLE my_staff
ADD CONSTRAINT fk_depart
FOREIGN KEY (depart_id)
REFERENCES my_depart(depart_id)

Table MY_STAFF altered.
Elapsed: 00:00:00.013