



Database Application

Table of Contents

Sr. No.	Description	Date	Page No	Signature
1	Practical 1: Data Definition Language a) Create b) Alter c) Drop d) Rename	09-09-2024		
2				
3				

Practical 1 - Introduction to Structures Query Language

1. Prerequisites

a. DDL - Data Definition Language:

Command	Description
CREATE	Creates a new table, a view of a table, or other objects in database
ALTER	Modifies an existing database object, such as a table.
DROP	Deletes an entire table, a view of a table or other object in the database.

b. DML - Data Manipulation Language:

Command	Description
INSERT	Creates a record
UPDATE	Modifies records.
DELETE	Deletes records.

c. DCL - Data Control Language:

Command	Description
GRANT	Gives a privilege to user.
REVOKE	Takes back privileges granted from User.

d. DQL - Data Query Language:

Command	Description
SELECT	Retrieves certain records from one or more tables.

2. To create a Table.

a. Syntax:

```
CREATE TABLE schema_name.table_name (
    column_1 data_type column_constraint,
    column_2 data_type column_constraint,
    ...
    table_constraint
);
```

3. Create a relation CUSTOMERS with the following attributes.

- a. ID of integer type and primary key.
- b. Name 20 characters.
- c. Age integer.
- d. Address 25 characters.
- e. Salary floating point 10.2 assign a default value 5000.

f. Solution Command:

```
CREATE TABLE customers_1 (
    ID int primary key,
    cname varchar(20),
    age int,
    Address varchar(20),
    salary number(10, 3) default 1000.00
);
```

g. Solution Output:

```
SQL> CREATE TABLE customers_1 (
  2      ID int primary key,
  3      cname varchar(20),
  4      age int,
  5      Address varchar(20),
  6      salary number(10, 3) default 1000.00
  7  );
Table created.
```

4. Create a table SUPPLIER.

- a. Supplier_id.
- b. Supplier_name.
- c. Contact_name.

d. Solution Command 1:

```
CREATE TABLE supplier_1 (
    supplier_id int primary key,
    supplier_name varchar(20),
    contact varchar(20)
);
```

e. Solution Output 1:

```
SQL> CREATE TABLE supplier_1 (
  2      supplier_id int primary key,
  3      supplier_name varchar(20),
  4      contact varchar(20)
  5  );
```

```
Table created.
```

f. Solution Command 2:

```
CREATE TABLE supplier_2 (
    supplier_id numeric(10),
    supplier_name varchar2(50),
    contact_name varchar2(50),
    CONSTRAINT supplier_pk_1 PRIMARY KEY(supplier_id)
);
```

g. Solution Output 2:

```
SQL> CREATE TABLE supplier_2 (
  2      supplier_id numeric(10),
  3      supplier_name varchar2(50),
  4      contact_name varchar2(50),
  5      CONSTRAINT supplier_1 PRIMARY KEY(supplier_id)
  6  );
```

```
Table created.
```

```
SQL>
```

5. Create a table SUPPLIER, but with COMPOSITE KEY.

a. Solution Command:

```
CREATE TABLE supplier_3 (
    supplier_id numeric(10) not null,
    supplier_name varchar2(50) not null,
    contact_name varchar2(50),
    CONSTRAINT supplier_pk_3 PRIMARY KEY(supplier_id, supplier_name)
);
```

b. Solution Output:

```
SQL> CREATE TABLE supplier_3 (
  2      supplier_id numeric(10) not null,
  3      supplier_name varchar2(50) not null,
  4      contact_name varchar2(50),
  5      CONSTRAINT supplier_pk_3 PRIMARY KEY(supplier_id, supplier_name)
  6  );
```

```
Table created.
```

```
SQL> -
```

6. Show the all the tables, which we've created above by using command DESCRIBE.

a. Solution Command 1:

```
DESCRIBE customers_1;
```

b. Solution Output 1:

```

SQL> DESCRIBE customers_1;
Name                                Null?    Type
-----                            -----
ID                                  NOT NULL NUMBER(38)
CNAME                             VARCHAR2(20)
AGE                               NUMBER(38)
ADDRESS                           VARCHAR2(20)
SALARY                           NUMBER(10,3)

SQL>

```

c. Solution Command 2:

```
DESCRIBE supplier_1;
```

d. Solution Output 2:

```

SQL> DESCRIBE supplier_1;
Name                                Null?    Type
-----                            -----
SUPPLIER_ID                         NOT NULL NUMBER(38)
SUPPLIER_NAME                        VARCHAR2(20)
CONTACT                            VARCHAR2(20)

SQL> -

```

e. Solution Command 3:

```
DESCRIBE supplier_2;
```

f. Solution Output 3:

```

SQL> DESCRIBE supplier_2;
Name                                Null?    Type
-----                            -----
SUPPLIER_ID                         NOT NULL NUMBER(10)
SUPPLIER_NAME                        VARCHAR2(50)
CONTACT_NAME                         VARCHAR2(50)

SQL> -

```

g. Solution Command 4:

```
DESCRIBE supplier_3;
```

h. Solution Output 4:

```

SQL> DESCRIBE supplier_3;
Name                                Null?    Type
-----                            -----
SUPPLIER_ID                         NOT NULL NUMBER(10)
SUPPLIER_NAME                        NOT NULL VARCHAR2(50)
CONTACT_NAME                         VARCHAR2(50)

SQL> -

```

7. Character Datatypes:

- a. CHAR(SIZE)
- b. VARCHAR
- c. VARCHAR2

8. Numeric Datatypes:

- a. INTEGER
- b. FLOAT

c. NUMBER(p, s)

d. DECIMAL

e. BOOLEAN

9. DATE/TIME Datatypes:

a. DATE

b. TIMESTAMP

10. Large Object (LOB) Datatypes:

a. BLOB

11. Create the table to store following details of students.

a. Roll Number - Integer

b. Name - 15 characters

c. Programme - 5 characters

d. Semester - 3 characters

e. Contact - 10 characters

f. Email - 20 characters

g. **Solution Command:**

```
CREATE TABLE students (
    roll_no int primary key,
    name varchar2(15),
    programme varchar2(5),
    semester varchar2(3),
    contact varchar2(10),
    email varchar2(20)
);
```

h. Solution Output:

```
SQL> CREATE TABLE students (
  2      roll_no int primary key,
  3      name varchar2(15),
  4      programme varchar2(5),
  5      semester varchar2(3),
  6      contact varchar2(10),
  7      email varchar2(20)
  8  );

Table created.

SQL> DESCRIBE students;
      Name          Null?    Type
-----+
ROLL_NO           NOT NULL  NUMBER(38)
NAME              VARCHAR(15)
PROGRAMME        VARCHAR(5)
SEMESTER         VARCHAR(3)
CONTACT          VARCHAR(10)
EMAIL             VARCHAR(20)

SQL> -
```

12. Explanation of CONSTRAINTS.

a. **NOT NULL:** Specifies that this column cannot hold NULL values (constraints of this type are not nameable).

b. **PRIMARY KEY:** Specifies the column that uniquely identifies a row in the table. The identified columns must be defined as **NOT NULL**.

c. **UNIQUE:** Specifies that values in the column must be unique.

d. **FOREIGN KEY:** Specifies that the values in the column must correspond to values in a referenced primary key or unique key column or that they are NULL.

e. **CHECK:** Specifies rules for values in the column.

13. Create EMPLOYEE table with following attributes and constraints.

a. ID (not null, number(5), primary key)

b. Empname (not null, varchar(20))

c. Address (varchar(30))

d. Age (>18, integer)

e. Salary (number (10, 2))

f. **Solution Command:**

```
CREATE TABLE EMPLOYEE_3 (
    emp_id number(5) not null,
    emp_name varchar2(20) not null,
    emp_address varchar2(30),
    age number(2),
    salary number(10, 2),
    CONSTRAINT emp_3_age_chk CHECK (age > 18),
    CONSTRAINT emp_3_pk PRIMARY KEY(emp_id)
);
```

g. **Solution Output:**

```
SQL> CREATE TABLE EMPLOYEE_3 (
  2      emp_id number(5) not null,
  3      emp_name varchar2(20) not null,
  4      emp_address varchar2(30),
  5      age number(2),
  6      salary number(10, 2),
  7      CONSTRAINT emp_3_age_chk CHECK (age > 18),
  8      CONSTRAINT emp_3_pk PRIMARY KEY(emp_id)
  9  );

Table created.

SQL> DESCRIBE EMPLOYEE_3
Name          Null?    Type
-----        -----
EMP_ID        NOT NULL NUMBER(5)
EMP_NAME      NOT NULL VARCHAR2(20)
EMP_ADDRESS   VARCHAR2(30)
AGE           NUMBER(2)
SALARY         NUMBER(10,2)

SQL> -
```

14. To Alter a Table.

a. **Syntax:**

```
ALTER TABLE table_name (
    ADD column_name column_definition;
)
```

15. Alter a customer table, which we've created above, and add email with datatype charater.

a. **Solution Command 1:**

```
ALTER TABLE customers_1
    ADD email varchar2(25);
```

b. **Solution Output 1:**

```

SQL> ALTER TABLE customers_1
  2      ADD email varchar2(25)
  3 ;

Table altered.

SQL> DESCRIBE customers_1
Name          Null?    Type
-----
ID           NOT NULL NUMBER(38)
CNAME        VARCHAR2(20)
AGE          NUMBER(38)
ADDRESS       VARCHAR2(20)
SALARY        NUMBER(10,3)
EMAIL         VARCHAR2(25)

SQL>

```

c. **Solution Command 2:**

```

ALTER TABLE customers_1
  ADD city varchar2(40) DEFAULT 'Seattle';

```

d. **Solution Output 2:**

```

SQL> ALTER TABLE customers_1
  2      ADD city varchar2(40) DEFAULT 'Seattle';

Table altered.

SQL> DESCRIBE customers_1
Name          Null?    Type
-----
ID           NOT NULL NUMBER(38)
CNAME        VARCHAR2(20)
AGE          NUMBER(38)
ADDRESS       VARCHAR2(20)
SALARY        NUMBER(10,3)
EMAIL         VARCHAR2(25)
CITY          VARCHAR2(40)

SQL>

```

16. **To alter a table by adding a multiple columns in a single commands.**

a. **Syntax:**

```

ALTER TABLE table_name
  ADD (
    column_name_1 column_definition,
    column_name_2 column_definition,
    column_name_3 column_definition,
    ...
    column_name_n column_definition,
  );

```

17. **Alter a customer table, which we've created above, and add multiple columns like contact number and pin code with datatype number and character respectively.**

a. **Solution Command 2:**

```

ALTER TABLE customers_1
  ADD (
    contact_num number(10),
    pin_code varchar2(50)
  );

```

b. **Solution Output 2:**

```

SQL> ALTER TABLE customers_1
  2      ADD (
  3          contact_num number(10),
  4          pin_code varchar2(50)
  5      );
Table altered.

SQL> DESCRIBE customers_1
Name           Null?    Type
-----          -----   -----
ID            NOT NULL NUMBER(38)
CNAME          VARCHAR2(20)
AGE             NUMBER(38)
ADDRESS         VARCHAR2(20)
SALARY          NUMBER(10,3)
EMAIL            VARCHAR2(25)
CITY             VARCHAR2(40)
CONTACT_NUM      NUMBER(10)
PIN_CODE         VARCHAR2(50)

SQL>

```

18. To Modify a Column in a Table.

a. Syntax:

```

ALTER TABLE table_name
MODIFY column_name column_type;

```

19. Modify the customer table, which we've created above, and increase the character length and also specify NOT NULL Constraints.

a. Solution Command 1:

```

ALTER TABLE customers_1
MODIFY cname varchar2(100) NOT NULL;

```

b. Solution Output 1:

```

SQL> ALTER TABLE customers_1
  2      MODIFY cname varchar2(100) NOT NULL;
Table altered.

SQL> DESCRIBE customers_1
Name           Null?    Type
-----          -----   -----
ID            NOT NULL NUMBER(38)
CNAME          NOT NULL VARCHAR2(100)
AGE             NUMBER(38)
ADDRESS         VARCHAR2(20)
SALARY          NUMBER(10,3)
EMAIL            VARCHAR2(25)
CITY             VARCHAR2(40)
CONTACT_NUM      NUMBER(10)
PIN_CODE         VARCHAR2(50)

SQL>

```

c. Solution Command 2:

```

ALTER TABLE customers_1
MODIFY city varchar2(75) DEFAULT 'Seattle' NOT NULL;

```

d. Solution Output 2:

```

SQL> ALTER TABLE customers_1
  2      MODIFY city varchar2(75) DEFAULT 'Seattle' NOT NULL;

Table altered.

SQL> DESCRIBE customers_1
Name          Null?    Type
-----        -----
ID           NOT NULL NUMBER(38)
CNAME        NOT NULL VARCHAR2(100)
AGE          NUMBER(38)
ADDRESS       VARCHAR2(20)
SALARY        NUMBER(10,3)
EMAIL         VARCHAR2(25)
CITY          NOT NULL VARCHAR2(75)
CONTACT_NUM   NUMBER(10)
PIN_CODE      VARCHAR2(50)

SQL>

```

20. To modify a table of multiple columns in a single commands.

a. Syntax:

```

ALTER TABLE table_name
MODIFY (
    column_name_1 column_type,
    column_name_2 column_type,
    column_name_3 column_type,
    ...
    column_name_n column_type,
);

```

21. Alter a customer table, which we've created above, and Modify multiple columns in it.

a. Solution Command:

```

ALTER TABLE customers_1
MODIFY (
    email varchar2(50) NOT NULL,
    contact_num NUMBER(12) NOT NULL
);

```

b. Solution Output:

```

SQL> ALTER TABLE customers_1
  2      MODIFY (
  3          email varchar2(50) NOT NULL,
  4          contact_num NUMBER(12) NOT NULL
  5      );

Table altered.

SQL> DESCRIBE customers_1
Name          Null?    Type
-----        -----
ID           NOT NULL NUMBER(38)
CNAME        NOT NULL VARCHAR2(100)
AGE          NUMBER(38)
ADDRESS       VARCHAR2(20)
SALARY        NUMBER(10,3)
EMAIL         VARCHAR2(50)
CITY          NOT NULL VARCHAR2(75)
CONTACT_NUM   NUMBER(12)
PIN_CODE      VARCHAR2(50)

SQL>

```

22. To Drop a Column from a Table.

a. Syntax:

```

ALTER TABLE table_name
DROP COLUMN column_name;

```

23. Drop the column from customer table, which we've created above, and Drop the column `"pin_code"`.

a. Solution Command:

```
ALTER TABLE customers_1
DROP COLUMN pin_code;
```

b. Solution Output:

```
SQL> ALTER TABLE customers_1
2      DROP COLUMN pin_code;

Table altered.

SQL> DESCRIBE customers_1
Name          Null?    Type
-----
ID           NOT NULL NUMBER(38)
CNAME        NOT NULL VARCHAR2(100)
AGE          NUMBER(38)
ADDRESS       VARCHAR2(20)
SALARY        NUMBER(10,3)
EMAIL         NOT NULL VARCHAR2(50)
CITY          NOT NULL VARCHAR2(75)
CONTACT_NUM   NOT NULL NUMBER(12)

SQL> -
```

24. To Rename any Column name from the Table.

a. Syntax:

```
ALTER TABLE table_name
RENAME COLUMN old_name TO new_name;
```

25. Rename the customer table, which we've created above, and Rename the column name `cname` to `customer_name`.

a. Solution Command:

```
ALTER TABLE customers_1
RENAME COLUMN cname TO customer_name;
```

b. Solution Output:

```
SQL> ALTER TABLE customers_1
2      RENAME COLUMN cname TO customer_name;

Table altered.

SQL> DESCRIBE customers_1;
Name          Null?    Type
-----
ID           NOT NULL NUMBER(38)
CUSTOMER_NAME NOT NULL VARCHAR2(100)
AGE          NUMBER(38)
ADDRESS       VARCHAR2(20)
SALARY        NUMBER(10,3)
EMAIL         NOT NULL VARCHAR2(50)
CITY          NOT NULL VARCHAR2(75)
CONTACT_NUM   NOT NULL NUMBER(12)

SQL>
```

26. To Rename the Table Name.

a. Syntax:

```
ALTER TABLE table_name
RENAME TO new_table_name;
```

27. Rename the customer table name.

a. Solution Command:

```
ALTER TABLE customers_1
    RENAME TO customers_1_1;
```

b. Solution Output:

```
SQL> ALTER TABLE customers_1
  2      RENAME TO customers_1_1;
Table altered.

SQL> DESCRIBE customers_1;
ERROR:
ORA-04043: object customers_1 does not exist

SQL> DESCRIBE customers_1_1;
Name          Null?    Type
-----        -----    -----
ID           NOT NULL NUMBER(38)
CUSTOMER_NAME NOT NULL VARCHAR2(100)
AGE          NUMBER(38)
ADDRESS       VARCHAR2(20)
SALARY        NUMBER(10,3)
EMAIL         NOT NULL VARCHAR2(50)
CITY          NOT NULL VARCHAR2(75)
CONTACT_NUM   NOT NULL NUMBER(12)

SQL>
```

28. To Drop the Constraint from the Table.

a. Syntax:

```
ALTER TABLE table_name
    DROP CONSTRAINT constraint_name;
```

29. Drop the Constraint from the supplier Table.

a. Solution Command:

```
ALTER TABLE supplier_3
    DROP CONSTRAINT supplier_pk_3;
```

b. Solution Output:

```
SQL> ALTER TABLE supplier_3
  2      DROP CONSTRAINT supplier_pk_3;
Table altered.

SQL> DESCRIBE supplier_3;
Name          Null?    Type
-----        -----    -----
SUPPLIER_ID   NOT NULL NUMBER(10)
SUPPLIER_NAME NOT NULL VARCHAR2(50)
CONTACT_NAME  VARCHAR2(50)

SQL>
```

30. To Add the Constraint in the Table.

a. Syntax:

```
ALTER TABLE table_name
    ADD CONSTRAINT constraint_name PRIMARY KEY (column1, column2, );
```

31. Add the Constraint in the supplier Table.

a. Solution Command:

```
ALTER TABLE supplier_3
    ADD CONSTRAINT supplier_pk_3 PRIMARY KEY (supplier_id, supplier_name);
```

b. Solution Output:

```
SQL> ALTER TABLE supplier_3
  2      ADD CONSTRAINT supplier_pk_3 PRIMARY KEY (supplier_id, supplier_name);
Table altered.

SQL> DESCRIBE supplier_3;
      Name          Null?    Type
-----  -----
SUPPLIER_ID           NOT NULL NUMBER(10)
SUPPLIER_NAME         NOT NULL VARCHAR2(50)
CONTACT_NAME          VARCHAR2(50)

SQL> -
```

32. To Disable Constraint in the Table.

a. Syntax:

```
ALTER TABLE table_name
  DISABLE CONSTRAINT constraint_name;
```

33. Disable the Constraint in the supplier Table.

a. Solution Command:

```
ALTER TABLE supplier_3
  DISABLE CONSTRAINT supplier_pk_3;
```

b. Solution Output:

```
SQL> ALTER TABLE supplier_3
  2      DISABLE CONSTRAINT supplier_pk_3;
Table altered.

SQL> DESCRIBE supplier_3;
      Name          Null?    Type
-----  -----
SUPPLIER_ID           NOT NULL NUMBER(10)
SUPPLIER_NAME         NOT NULL VARCHAR2(50)
CONTACT_NAME          VARCHAR2(50)

SQL>
```

34. To Enable Constraint in the Table.

a. Syntax:

```
ALTER TABLE table_name
  ENABLE CONSTRAINT constraint_name;
```

35. Enable the Constraint in the supplier Table.

a. Solution Command:

```
ALTER TABLE supplier_3
  ENABLE CONSTRAINT supplier_pk_3;
```

b. Solution Output:

```

SQL> ALTER TABLE supplier_3
  2      ENABLE CONSTRAINT supplier_pk_3;

Table altered.

SQL> DESCRIBE supplier_3;
   Name          Null?    Type
   -----
SUPPLIER_ID           NOT NULL NUMBER(10)
SUPPLIER_NAME        NOT NULL VARCHAR2(50)
CONTACT_NAME         VARCHAR2(50)

SQL> A_

```

36. Exercise:

- Add new column credit to customers table.

a. Solution Command:

```

ALTER TABLE customers_1_1
  ADD credit varchar2(25);

```

b. Solution Output:

```

SQL> ALTER TABLE customers_1_1
  2      ADD credit varchar2(25);

Table altered.

SQL> DESCRIBE customers_1_1;
   Name          Null?    Type
   -----
ID            NOT NULL NUMBER(38)
CUSTOMER_NAME        NOT NULL VARCHAR2(100)
AGE           NUMBER(38)
ADDRESS        VARCHAR2(20)
SALARY         NUMBER(10,3)
EMAIL          VARCHAR2(50)
CITY           VARCHAR2(75)
CONTACT_NUM    NOT NULL NUMBER(12)
CREDIT         VARCHAR2(25)

SQL> -

```

- Add two columns contact and email to customer table.

i. Solution Command:

```

ALTER TABLE supplier_3
  ADD (
    contact_num number(10),
    email varchar2(50)
  );

```

ii. Solution Output:

```
SQL> ALTER TABLE supplier_3
  2      ADD (
  3          contact_num number(10),
  4          email varchar2(50)
  5      );

Table altered.

SQL> DESCRIBE supplier_3;
Name           Null?    Type
-----          -----
SUPPLIER_ID        NOT NULL NUMBER(10)
SUPPLIER_NAME      NOT NULL VARCHAR2(50)
CONTACT_NAME       VARCHAR2(50)
CONTACT_NUM        NUMBER(10)
EMAIL             VARCHAR2(50)

SQL>
```