CREATE TABLE customers

( customer\_id number(10) NOT NULL,

customer\_name varchar2(50) NOT NULL,

city varchar2(50)

);

CREATE TABLE customers

( customer\_id number(10) NOT NULL,

customer\_name varchar2(50) NOT NULL,

city varchar2(50),

CONSTRAINT customers\_pk PRIMARY KEY (customer\_id)

);

**Practice Exercise #1:**

Create an Oracle table called *suppliers* that stores supplier ID, name, and address information.

Solution for Practice Exercise #1:

The Oracle CREATE TABLE statement for the *suppliers* table is:

CREATE TABLE suppliers

( supplier\_id number(10) NOT NULL,

supplier\_name varchar2(50) NOT NULL,

address varchar2(50),

city varchar2(50),

state varchar2(25),

zip\_code varchar2(10)

);

## Practice Exercise #2:

Create an Oracle table called customers that stores customer ID, name, and address information.

But this time, the customer ID should be the [primary key](https://www.techonthenet.com/oracle/primary_keys.php) for the table.

### Solution for Practice Exercise #2:

The Oracle CREATE TABLE statement for the customers table is:

CREATE TABLE customers

( customer\_id number(10) NOT NULL,

customer\_name varchar2(50) NOT NULL,

address varchar2(50),

city varchar2(50),

state varchar2(25),

zip\_code varchar2(10),

CONSTRAINT customers\_pk PRIMARY KEY (customer\_id)

);

CREATE TABLE employees

( employee\_number number(10) NOT NULL,

employee\_name varchar2(50) NOT NULL,

department\_id number(10),

salary number(6),

CONSTRAINT employees\_pk PRIMARY KEY (employee\_number),

CONSTRAINT fk\_departments

FOREIGN KEY (department\_id)

REFERENCES departments(department\_id)

);

**Character Datatypes**

The following are the **Character Datatypes** in Oracle/PLSQL:

| Data Type Syntax | Oracle 9i | Oracle 10g | Oracle 11g | Explanation |
| --- | --- | --- | --- | --- |
| char(size) | Maximum size of 2000 bytes. | Maximum size of 2000 bytes. | Maximum size of 2000 bytes. | Where ***size*** is the number of characters to store. Fixed-length strings. Space padded. |
| nchar(size) | Maximum size of 2000 bytes. | Maximum size of 2000 bytes. | Maximum size of 2000 bytes. | Where ***size*** is the number of characters to store. Fixed-length NLS string Space padded. |
| nvarchar2(size) | Maximum size of 4000 bytes. | Maximum size of 4000 bytes. | Maximum size of 4000 bytes. | Where ***size*** is the number of characters to store. Variable-length NLS string. |
| varchar2(size) | Maximum size of 4000 bytes.  Maximum size of 32KB in PLSQL. | Maximum size of 4000 bytes.  Maximum size of 32KB in PLSQL. | Maximum size of 4000 bytes.  Maximum size of 32KB in PLSQL. | Where ***size*** is the number of characters to store. Variable-length string. |
| long | Maximum size of 2GB. | Maximum size of 2GB. | Maximum size of 2GB. | Variable-length strings. (backward compatible) |
| raw | Maximum size of 2000 bytes. | Maximum size of 2000 bytes. | Maximum size of 2000 bytes. | Variable-length binary strings |
| long raw | Maximum size of 2GB. | Maximum size of 2GB. | Maximum size of 2GB. | Variable-length binary strings. (backward compatible) |

**Numeric Datatypes**

The following are the **Numeric Datatypes** in Oracle/PLSQL:

| Data Type Syntax | Oracle 9i | Oracle 10g | Oracle 11g | Explanation |
| --- | --- | --- | --- | --- |
| number(p,s) | Precision can range from 1 to 38. Scale can range from -84 to 127. | Precision can range from 1 to 38. Scale can range from -84 to 127. | Precision can range from 1 to 38. Scale can range from -84 to 127. | Where ***p*** is the precision and ***s*** is the scale.  For example, number(7,2) is a number that has 5 digits before the decimal and 2 digits after the decimal. |
| numeric(p,s) | Precision can range from 1 to 38. | Precision can range from 1 to 38. | Precision can range from 1 to 38. | Where ***p*** is the precision and ***s*** is the scale.  For example, numeric(7,2) is a number that has 5 digits before the decimal and 2 digits after the decimal. |
| float |  |  |  |  |
| dec(p,s) | Precision can range from 1 to 38. | Precision can range from 1 to 38. | Precision can range from 1 to 38. | Where ***p*** is the precision and ***s*** is the scale.  For example, dec(3,1) is a number that has 2 digits before the decimal and 1 digit after the decimal. |
| decimal(p,s) | Precision can range from 1 to 38. | Precision can range from 1 to 38. | Precision can range from 1 to 38. | Where ***p*** is the precision and ***s*** is the scale.  For example, decimal(3,1) is a number that has 2 digits before the decimal and 1 digit after the decimal. |
| integer |  |  |  |  |
| int |  |  |  |  |
| smallint |  |  |  |  |
| real |  |  |  |  |
| double precision |  |  |  |  |

**Date/Time Datatypes**

The following are the **Date/Time Datatypes** in Oracle/PLSQL:

| Data Type Syntax | Oracle 9i | Oracle 10g | Oracle 11g | Explanation |
| --- | --- | --- | --- | --- |
| date | A date between Jan 1, 4712 BC and Dec 31, 9999 AD. | A date between Jan 1, 4712 BC and Dec 31, 9999 AD. | A date between Jan 1, 4712 BC and Dec 31, 9999 AD. |  |
| timestamp (*fractional seconds precision*) | ***fractional seconds precision*** must be a number between 0 and 9. (default is 6) | ***fractional seconds precision*** must be a number between 0 and 9. (default is 6) | ***fractional seconds precision*** must be a number between 0 and 9. (default is 6) | Includes year, month, day, hour, minute, and seconds.  For example: timestamp(6) |
| timestamp (*fractional seconds precision*) with time zone | ***fractional seconds precision*** must be a number between 0 and 9. (default is 6) | ***fractional seconds precision*** must be a number between 0 and 9. (default is 6) | ***fractional seconds precision*** must be a number between 0 and 9. (default is 6) | Includes year, month, day, hour, minute, and seconds; with a time zone displacement value.  For example: timestamp(5) with time zone |
| timestamp (*fractional seconds precision*) with local time zone | ***fractional seconds precision*** must be a number between 0 and 9. (default is 6) | ***fractional seconds precision*** must be a number between 0 and 9. (default is 6) | ***fractional seconds precision*** must be a number between 0 and 9. (default is 6) | Includes year, month, day, hour, minute, and seconds; with a time zone expressed as the session time zone.  For example: timestamp(4) with local time zone |
| interval year (*year precision*) to month | ***year precision*** is the number of digits in the year. (default is 2) | ***year precision*** is the number of digits in the year. (default is 2) | ***year precision*** is the number of digits in the year. (default is 2) | Time period stored in years and months.  For example: interval year(4) to month |
| interval day (*day precision*) to second (*fractional seconds precision*) | ***day precision*** must be a number between 0 and 9. (default is 2)  ***fractional seconds precision*** must be a number between 0 and 9. (default is 6) | ***day precision*** must be a number between 0 and 9. (default is 2)  ***fractional seconds precision*** must be a number between 0 and 9. (default is 6) | ***day precision*** must be a number between 0 and 9. (default is 2)  ***fractional seconds precision*** must be a number between 0 and 9. (default is 6) | Time period stored in days, hours, minutes, and seconds.  For example: interval day(2) to second(6) |

**Large Object (LOB) Datatypes**

The following are the **LOB Datatypes** in Oracle/PLSQL:

| Data Type Syntax | Oracle 9i | Oracle 10g | Oracle 11g | Explanation |
| --- | --- | --- | --- | --- |
| bfile | Maximum file size of 4GB. | Maximum file size of 232-1 bytes. | Maximum file size of 264-1 bytes. | File locators that point to a binary file on the server file system (outside the database). |
| blob | Store up to 4GB of binary data. | Store up to (4 gigabytes -1) \* (the value of the CHUNK parameter of LOB storage). | Store up to (4 gigabytes -1) \* (the value of the CHUNK parameter of LOB storage). | Stores unstructured binary large objects. |
| clob | Store up to 4GB of character data. | Store up to (4 gigabytes -1) \* (the value of the CHUNK parameter of LOB storage) of character data. | Store up to (4 gigabytes -1) \* (the value of the CHUNK parameter of LOB storage) of character data. | Stores single-byte and multi-byte character data. |
| nclob | Store up to 4GB of character text data. | Store up to (4 gigabytes -1) \* (the value of the CHUNK parameter of LOB storage) of character text data. | Store up to (4 gigabytes -1) \* (the value of the CHUNK parameter of LOB storage) of character text data. | Stores [Unicode](https://www.techonthenet.com/unicode/chart.php) data. |

DROP TABLE customers;

CREATE TABLE supplier

(

supplier\_id numeric(10) not null,

supplier\_name varchar2(50) not null,

contact\_name varchar2(50),

CONSTRAINT supplier\_pk PRIMARY KEY (supplier\_id, supplier\_name)

);

Create table Location

(

Location\_id Number(20) PRIMARY KEY,

City Varchar(20)

);

Create Table Job

(

job\_id Number(20) PRIMARY KEY,

Designation Varchar(20)

);

Create table Employee

(

ID Number(20) Primary key ,

Last\_Name Varchar(20),

First\_Name Varchar(20),

Middle\_Name Varchar(20),

Job\_id Number(20),

Manager\_id Number(20),

Hire\_Date date,

Salary Number(20),

Comm Number(20),

Department\_id Number(20),

Foreign Key(Job\_id)REFERENCES JOB(Job\_id),

Foreign Key(Manager\_id)REFERENCES Employee(ID),

Foreign Key(Department\_id) REFERENCES Department(Department\_id)

);

Insert into Location Values(122,'New York');

Insert into Location Values(123,'Dallas');

Insert into Location Values(124,'Chicago');

Insert into Location Values(167,'Boston');

Select \* from Location;

insert into Job values(667,'Clerk');

Insert into Job values(668,'Staff');

Insert into Job values(669,'Analyst');

Insert into Job values(670,'Sales Person');

Insert into job values(671,'Manager');

Insert into Job values(672,'President');

select \* from Job;

create table toys\_csv (

toy\_name varchar2(10),

  weight number,

  colour varchar2(10)

) organization external (

default directory ext\_files

  location ( 'toys.csv' )

);

1. **NOT NULL**: This constraint ensures that a query does not insert a NULL value in a column on which the constraint has been created.
2. **UNIQUE**: This constraint ensures that the query inserts only unique values in the column on which the constraint has been created.
3. **CHECK**: When we insert a record in a table, the check constraint ensures that the values that we are inserting in a column must satisfy the condition specified in a CHECK constraint.
4. **DEFAULT**: The DEFAULT constraint inserts a default value. If we do not specify the values for the column on which the default constraint, it inserts the default value specified in the DEFAULT constraints.
5. **PRIMARY KEY**: The Primary key constraint is used to identify each record uniquely. Following are the characteristics of the Primary key
   1. The primary key consists of one or more columns.
   2. The primary key contains unique values and cannot contain a NULL value.
   3. We can create one primary key in a table.
   4. When we create a primary key, a clustered index will be created automatically.
6. **FOREIGN KEY**: The foreign key is a column that creates a link between two tables. The foreign key of any table refers to the Primary Key of another table. A table can have one or more foreign keys. A foreign key constraint prevents the UPDATE and DELETE statement that breaks the link between two tables.

password