Create Table

```
CREATE TABLE suppliers
( supplier_id number(10) NOT NULL,
supplier_name varchar2(50) NOT NULL,
city varchar2(50)
);


CREATE TABLE suppliers
( supplier_id number(10) PRIMARY KEY NOT NULL,
supplier_name varchar2(50) NOT NULL,
city varchar2(50)
);


CREATE TABLE suppliers
( supplier_id number(10) NOT NULL,
supplier_name varchar2(50) NOT NULL,
city varchar2(50),
PRIMARY KEY(supplier_id)
);
```

Create Table

- CREATE TABLE suppliers
- (supplier_id number(10) NOT NULL,
- supplier_name varchar2(50) NOT NULL,
- city varchar2(50),
- CONSTRAINT suppliers_id_pk PRIMARY KEY (supplier_id)
-);
-

- CREATE TABLE suppliers
- (supplier_id number(10) NOT NULL,
- supplier_name varchar2(50) NOT NULL,
- city varchar2(50),
- CONSTRAINT suppliers_id_name_pk PRIMARY KEY (supplier_id, supplier_name)
-);

Insert Rows

- INSERT INTO suppliers
- (supplier_id, supplier_name)
- VALUES
- (1000, 'Apple');
-
- **OR**
-
- INSERT INTO suppliers VALUES
- (2000, 'Facebook','');
-
- **OR**
-
- INSERT ALL
- INTO suppliers (supplier_id, supplier_name) VALUES (3000, 'IBM')
- INTO suppliers (supplier_id, supplier_name) VALUES (4000, 'Microsoft')
- INTO suppliers (supplier_id, supplier_name) VALUES (5000, 'Google')
- SELECT * FROM dual;

Drop, Rename Table

- **Truncate Table**
-
- TRUNCATE TABLE suppliers;
-
- **Drop Table**
-
- DROP TABLE suppliers;
-
- **Rename Table**
-
- ALTER TABLE suppliers RENAME to suppliers_new;
- **OR**
- RENAME suppliers to suppliers_new;
-
- **Rename Column**
- ALTER TABLE suppliers
- RENAME COLUMN city to place;

Add, Drop Columns

- **Add Column**
- ALTER TABLE suppliers
- ADD address varchar2(100);
- **OR**
- ALTER TABLE suppliers
- ADD (state varchar2(100),
- email varchar2(100));
-
- **Drop Column**
- ALTER TABLE suppliers
- DROP COLUMN address;
- **OR**
- ALTER TABLE suppliers
- DROP (state , email);
-

- **Modify Column**
- ALTER TABLE suppliers
- MODIFY (supplier_name varchar2(100));
- **OR**
- ALTER TABLE suppliers
- MODIFY (supplier_name varchar2(100),
- place varchar2(75));

Primary Key Constraint

- **Primary Key Constraint**

- ALTER TABLE suppliers
- ADD PRIMARY KEY (supplier_id);
- **OR**
- ALTER TABLE suppliers
- ADD CONSTRAINT suppliers_id_pk PRIMARY KEY (supplier_id);
- **OR**
- ALTER TABLE suppliers
- ADD CONSTRAINT suppliers_id_name_pk PRIMARY KEY (supplier_id, supplier_name);
-

Drop Primary Key Constraint

ALTER TABLE suppliers
DROP PRIMARY KEY;
OR
ALTER TABLE suppliers
DROP CONSTRAINT
suppliers_id_name_pk;

Check Constraint

- **Check Constraint**
- A **check constraint** allows you to specify a condition on each row in a table.
-
- Check Constraint Using Create Table
- CREATE TABLE suppliers
- (supplier_id number(10) not null,
- supplier_name varchar2(50) not null,
- CONSTRAINT check_supplier_id CHECK (supplier_id BETWEEN 100 and 9999)
-);

Enable, Disable Constraint

- **Add Constraint**
- ALTER TABLE suppliers
- ADD CONSTRAINT check_supplier_name
- CHECK (supplier_name IN ('IBM', 'MICROSOFT', 'GOOGLE'));
-
- **Drop Constraint**
- ALTER TABLE suppliers
- DROP CONSTRAINT check_supplier_name;
-
- **Disable Constraint**
- ALTER TABLE suppliers
- DISABLE CONSTRAINT check_supplier_name;
-

Enable Constraint

```
ALTER TABLE suppliers
ENABLE CONSTRAINT
check_supplier_name;
```

Not Null Constraint

```
ALTER TABLE suppliers
MODIFY (supplier_name NOT NULL);
```

Unique Constraint

```
ALTER TABLE suppliers
ADD CONSTRAINT supplier_city
UNIQUE (city);
```

Defining a Constraint

- A constraint can be created at the same time the table is created, or it can be added to the table afterward. There are two levels where a constraint is defined:
  - Column level.
  - Table level.

Column level

- A column-level constraint references a single column and is defined along with the definition of the column.

- Any constraint can be defined at the column level except for a FOREIGN KEY and COMPOSITE primary key constraints.

Column datatype [CONSTRAINT constraint_name] constraint_type

Example:

Building VARCHAR2(7) CONSTRAINT location_building_nn NOT NULL

Table level

- A table-level constraint references one or more columns and is defined separately from the definitions of the columns.
- Normally, it is written after all columns are defined.
- All constraints can be defined at the table level except for the NOT NULL constraint.

[CONSTRAINT constraint_name] constraint_typ (Column, . . .),

Example:

CONSTRAIN location_roomid_pk PRIMARY KEY(Roomid)

The Primary Key Constrain

- The PRIMARY KEY constraint is also known as the **entity integrity constraint**

- It creates a primary key for the table. A table can have only one primary key constraint.

- If a table uses more than one column as its primary key (i.e., a composite key), the key can only be declared at the table level.

The Primary Key Constrain

- At the column level, the constraints is defined by

  DeptId NUMBER (2) CONSTRAINT dept_deptid_pk PRIMARY KEY,

- At the table level, the constraint is defined by

  CONSTRAINT dept_deptid_pk PRIMARY KEY(DeptId),

Composite key

- Syntax

```
CONSTRAINT constraint_name
PRIMARY KEY (columnname1, columnname2 …)
```

- Example:

```
CREATE TABLE enrollment
(s_id NUMBER(5) CONSTRAINT enrollment_s_id_fk REFERENCES student(s_id),
c_sec_id NUMBER(8) CONSTRAINT enrollment_c_sec_id_fk REFERENCES
course_section(c_sec__id),
CONSTRAINT enrollment_s_id_c_sec_id_pk PRIMARY KEY (s_id, c_sec_id));
```

The FOREIGN KEY Constraint

- The FOREIGN KEY constraint is also known as the **referential integrity constraint.**

- It uses a column or columns as a foreign key, and it establishes a relationship with the primary key of the same or another table.

The FOREIGN KEY Constraint

- To establish a foreign key in a table, the other referenced table and its primary key must already exist.

- Foreign key and referenced primary key columns need not have the same name, but a foreign key value **must match** the value in the parent table's primary key value or be NULL

The FOREIGN KEY Constraint

- At the table level **ONLY**

CONSTRAINT student_facultyid_fk FOREIGN KEY(FacultyId)

REFERENCES faculty (FacultyId),

The FOREIGN KEY Constraint

- Foreign key
  - Syntax (**placed at end of table definition**)

```
CONSTRAINT constraint_name
FOREIGN KEY (columnname)
REFERENCES primary_key_tablename
  (primary_key_columnname)
```

  - Example of foreign key defined in the Faculty table:

```
CONSTRAINT faculty_loc_id_fk
FOREIGN KEY (loc_id)
REFERENCES location (loc_id)
```

The FOREIGN KEY Constraint

- Foreign key (continued)
  - Syntax (**placed within table definition**)

```
CONSTRAINT constraint_name
REFERENCES primary_key_tablename
(primary_key_columnname)
```

  - Example:

```
loc_id NUMBER(6) CONSTRAINT faculty_loc_id_fk
REFERENCES location (loc_id)
```

The NOT NULL Constraint

- The NOT NULL constraint ensures that the column has a value and the value is not a null value
- A space or a numeric zero is not a null value
- At the column level **ONLY**, the constraint is defined by:

Name VARCHAR2(15) CONSTRAINT faculty_name_nn NOT NULL,

The UNIQUE Constraint

- The UNIQUE constraint requires that every value in a column or set of columns be unique.

- At the table level, the constraint is defined by

  CONSTRAINT dept_deptname_uk UNIQUE(DeptName),

- At the column level, the constraint is defined by:

  DeptName VARCHAR2(12) CONSTRAINT dept_deptname_uk UNIQUE,

The CHECK Constraint

- The CHECK constraint defines a condition that every row must satisfy

- At the column level, the constraint is defined by
  DeptId NUMBER(2) CONSTRAINT dept_deptid_cc
  CHECK((DeptId >= 10) and (DeptId <= 99)),

- At the table level, the constraint is defined by:
  CONSTRAINT dept_deptid_cc
  CHECK((DeptId >= 10) and (DeptId <= 99)),