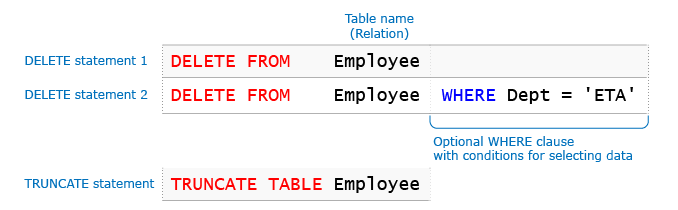
**PF**

Call by Value--- operations will not affect the original value

Call by reference--- operation done will change the original value

def **f1**(par1):

par1 = par1+[30]

print(*"inside the function"*, par1)

val1=[10,20]

f1(val1)

print(*"outside the function"*, val1)

inside the function [10, 20, 30]

outside the function [10, 20]

def **f1**(par1):

par1 += [30]

print(*"inside the function"*, par1)

val1=[10,20]

f1(val1)

print(*"outside the function"*, val1)

inside the function [10, 20, 30]

outside the function [10, 20, 30]

In += , 30 was added to the reference of val1

In par1=par1+[30],

In python

.append always goes to reference of the list thus editing the original one

list1+= [23]

also updates the original

but

list1= list1+[23] does not edit the reference it makes a new list

Global and local variables

var1=100

def **f2**():

global var1

print(var1)

var1=50

print(var1)

print(*"out 1"*,var1)

f2()

print(*"out 2"*, var1)

outputs:

out 1 100

100

50

out 2 50

---thus global variable when declared inside was changed to 50---

import datetime

print(datetime.date.today().strftime("%d/%m/%Y"))

%a weekday short

%A weekday long

%w weekday as a no 0sunday 6 Saturday

%d day of month 01-31

%b month name short

%B month name full

%m month as a no

%y year short

%Y year full

%H hour 00-23

open

**OOP**

in python

Inheritance: when an object is another object, inherits properties

Aggregation: customer has an address, and address will continue to exist if customer is deleted

Composition: College has a department. If the college closes, the department is also closed.

Inheritance:

* All properties are inherited
* Private properties of the parent class cannot be accessed from the child class
* Although private properties can be accessed as for example self.\_Animal\_\_eyes
* Call super.\_\_init\_\_() in the constructer to pass on properties to the super constructor and inherit them
* The overridden method of a super class can be called simply by going super().method\_name()

Or in case of multiple inheritance, ClassName.method\_name() can be called to access the method of any specific parent class.

* MULTIPLE Inheritance—super() is a reference to both the parents, when calling super().method\_name, the method is searched first in the first parent(one declared to left side), and the if it is not fond, it looks for the method in the second parent

To call a method from specific parent class, ClassName.method\_name() can be called.

**DATA STRUCTURES**

**Algorithm steps:**

**add(element)**:

1. When the list is initially created, it is created

with a certain capacity.

2. While adding the elements, if the list is filled

to the capacity,

a. Create a new list with increased capacity

b. Copy the elements of initial list to the new list

3. Add the element to the end of the existing elements

in the list

|  |  |
| --- | --- |
| http://ilp/fp2/Generic/DSA/images/bwd.png  **Looking back** | **append()** method of **Python list** implicitly implements the above algorithm. |

Quite a slow implementation

Sort a list in python according to key

def function(x):

return x[0]

List1.sort(key=function)

Or list1.sort(key=lambda x: x[0])

Dictionary update function:

**\*\***Updates the key, if the key does not exist, adds\*\*

Dictionary implements hashing for its internal working

**SQL**

**Primary key**

Non null

Must be unique

Must identify unique row

Can be a single or a combination of columns

Primary key is the id of any single document or as called in sql it is the id of any row in a table.

**Candidate key**

Must id a unique row

Can be null

Single or multiple cols

A primary key is a candidate key.

**Foreign key**

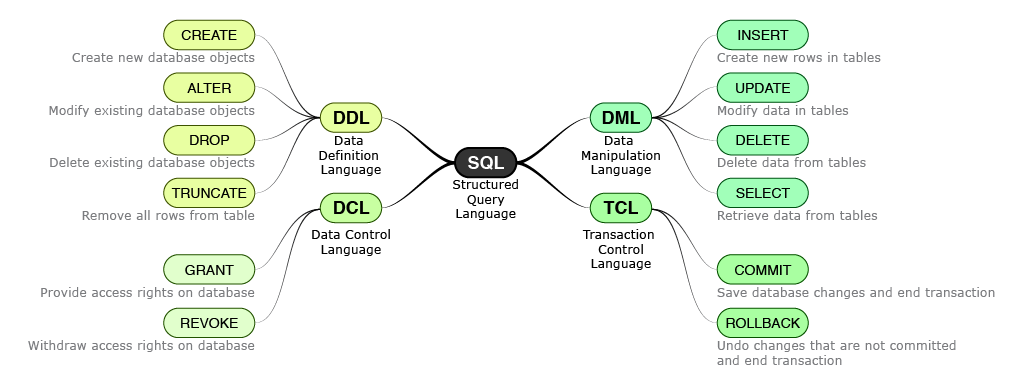
Can be null

Can be duplicate

Foreign key in child must present in parent table

Foreign key is the lookup field, from another table

It has to be present in the other table that is being linked.



**DATA TYPES**

* Blob- binary large
* Clob- characters large
* Date dd-mon-yy
* Timestamp- date + time (accurate)
* Number (6,2) implies 5 total digits and only 2 to the right of the decimal pt. if there are more to the right of the decimal pt then it is rounded off, 12.78 goes to 12.8 and 12.73 to 12.7

**Create tables:**

Gving constraints:

In front of a column name write

c **char**(2) **CONSTRAINT** onc **CHECK**(c **IN** ('a', 'b', 'c'))

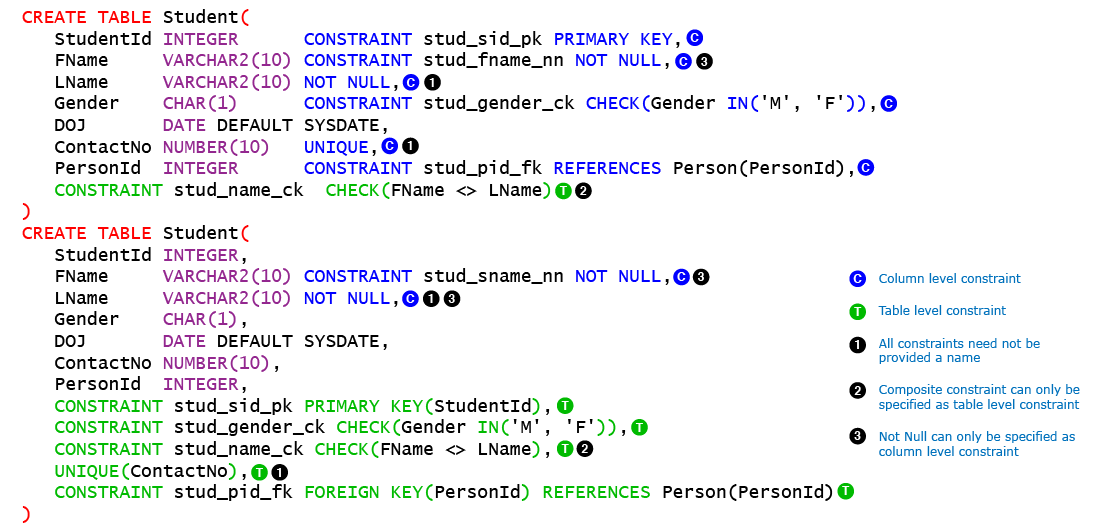
Diff constraints such as unique, not null, primary key, foreign key, default, index etc exist

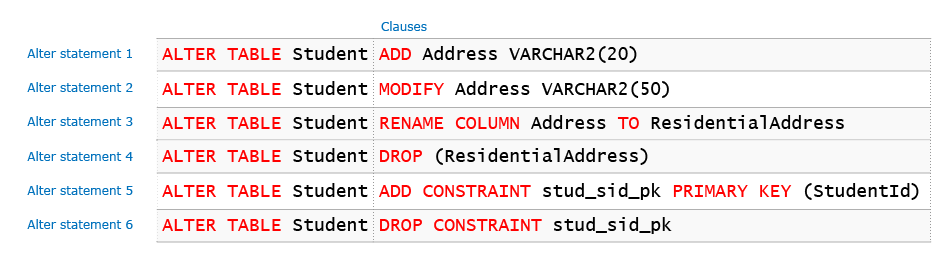
Primary key can be straight away declared:

d **integer** **PRIMARY** **KEY** ,

foreign key can be directly given as references..

table2a **integer** **REFERENCES** table2\_adwait22(a)





Constraints:

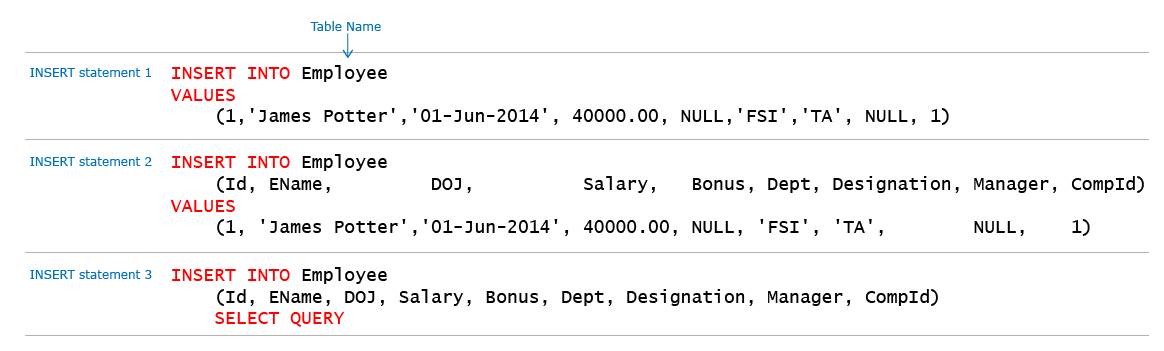
Unique: null values are allowed and non null values must be unique

|  |  |  |  |
| --- | --- | --- | --- |
| Equal to | = | 15 = 5 | false |
| Not equal to | <> | 15 <> 5 | true |
| Greater than | > | 15 > 5 | true |
| Greater than equal to | >= | 15 >= 5 | true |
| Less than | < | 15 < 5 | false |
| Less than equal to | <= | 15 <= 5 | false |

**constraint** arc1 **check**(**substr**(arcode, 1, 1)='A')

substr function to use to check whether certain column substr matches or not.

**Insert**

****

**Simple queries:**

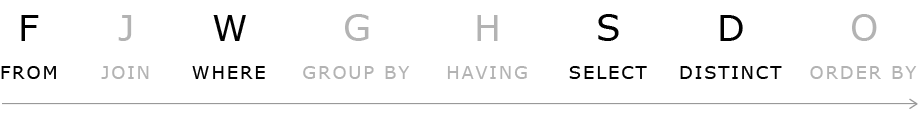
SELECT ID, ENAME FROM Employee WHERE ID IN (2,3)

SELECT ID, ENAME FROM Employee WHERE ID NOT IN (2,3)

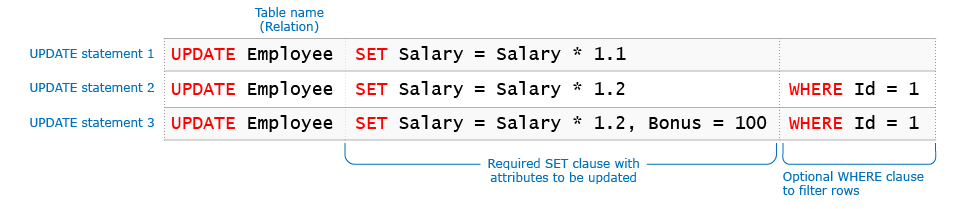
Where col\_name Is null

Where col\_name is not null

A query is **executed** in the following fasion:



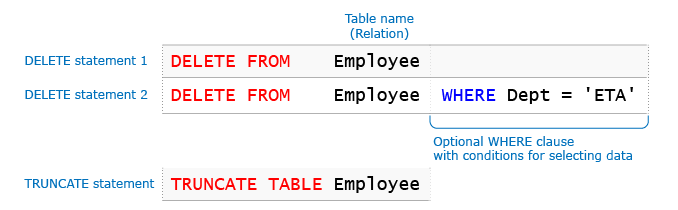
Update query:

****

**Concat** function to concatenate strings can be used in update query

**Substr** fn to get substring.

DELETE



**Error codes documentation:**

<http://docs.oracle.com/cd/E11882_01/server.112/e17766/toc.htm>

|  |  |  |
| --- | --- | --- |
| ABS | ABS(value) | Returns absolute value of a number |
| ROUND | ROUND(value, digits) | Rounds the value to specified decimal digits |
| CEIL | CEIL(value) | Rounds up the fractional value to next integer |
| FLOOR | FLOOR(value) | Rounds down the fractional value to the lower integer |

|  |  |  |
| --- | --- | --- |
| UPPER | UPPER(value) | Converts value to upper case |
| LOWER | LOWER(value) | Converts value in lower case |
| CONCAT | CONCAT(value1, value2) | Concatenates value1 and value2 |
| LENGTH | LENGTH(value) | Returns the number of characters in value. |

|  |  |  |
| --- | --- | --- |
| TO\_CHAR | TO\_CHAR(value, format) | Converts a number or a date to a string. Use this function for formatting dates and numbers. |
| TO\_DATE | TO\_DATE (value, format) | Converts a string to a date. |
| TO\_NUMBER | TO\_NUMBER (value, format) | Converts a string to a number. |

**Aggregate functions** or Multiple row functions **min,max, avg, count, sum, count with distinct.**

SELECT TO\_CHAR(Accdt,'dd')"DAY", TO\_CHAR(Accdt,'mm')"MON" FROM Customer

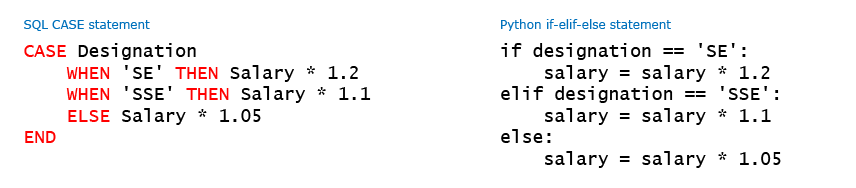
When accdt is ’30-jan-19’ , TO\_CHAR(accdt, ‘dd’) returns the day of the date, ‘mm’ returns month

|  |  |  |
| --- | --- | --- |
| NVL | NVL(value1, value2) | Substitutes value1 by value2 if value1 is NULL. The data type of value1 and value2 must be same. |
| USER | USER | Returns the current logged in user |

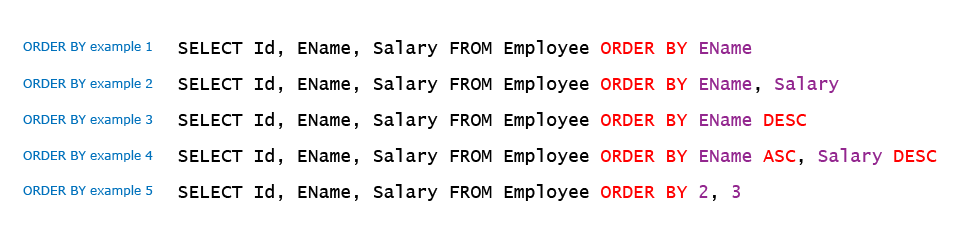
Dual is a single line table that exists always

It is a dummy table

|  |  |  |
| --- | --- | --- |
| SYSDATE | SYSDATE | Returns current date of System i.e. the host on which database server is installed. |
| SYSTIMESTAMP | SYSTIMESTAMP | Returns current timestamp of the System. |
| ADD\_MONTHS | ADD\_MONTHS(date, n) | Adds n months to the given date. |
| MONTHS\_BETWEEN | MONTHS\_BETWEEN(date1,date2) | Finds difference between two dates in months. |



orderby



Null is treated as the last records in orderby asc and first in orderby desc



**to\_char (sldate, 'Month')**