Differentiate Between

Type Casting ***(Explicit Conversion)*** and Automatic type conversion ***(Implicit Conversion)***

Type Casting(Explicit Conversion)	Automatic Type Conversion(Implicit Conversion)
It is an explicit process of conversion of a data from one type to another. (It is performed by the programmer.)	It is an implicit process of conversion of a data from one data type to another. It is performed by the compiler. It is also called as type promotion.

e.g.
k=123.456
i= int(k)
	e.g.
	i=2.3;
	j=10
	i=j

Explicit conversion do require cast operator to perform conversion.	Implicit conversion do not require any special syntax.
In explicit conversion, data loss may or may not be take place during data conversion. Hence there is a risk of information loss.	In Implicit conversion, no data loss take place during the data conversion.

Arithmetic Assignment Operator	**Assignment**
Used to assign values to the variables after performing arithmetic operations.	**Used to assign values to the variables.**
respresented by (+=,-=,*=,/=,%=,//=)	**Represented by (=)**

Global variable	**Local Variable**
Global variables are defined outside of all the functions, generally on top of the program.	A local variable is declared within the body of a function or a block
The global variables will hold their value throughout the life-time of your program.	Local variable only use within the function or block where it is declare.

a=10   //global variable
def fun():
   b=20
   print(b)   //local variable
print(a)	

What do you understand by local and global scope of variables? How can you access a global variable inside the function, if function has a variable with same name.

Ans. A global variable is a variable that is accessible globally. A local variable is one that is only accessible to the current scope, such as temporary variables used in a single function definition.
A variable declared outside of the function or in global scope is known as global variable. This means, global variable can be accessed inside or outside of the function where as local variable can be used only inside of the function. We can access by declaring variable as global A.

Actual and formal parameter/argument

Actual parameter/argument	Formal parameter/argument
The arguments that are passed in a function call are called actual arguments.	The formal arguments are the parameters/arguments in a function declaration.
These arguments are defined in the calling function.	The scope of formal arguments is local to the function definition in which they are used.

```
def sum(i, j, k):                    //called function //formal parameter/ arguments
   s = i + j + k;
   print(s)

a = 5;
sum(3, 2 , a);                       //calling function // actual parameter /arguments
```

3,2, a are actual arguments   and   i, j, k are formal arguments.

Default argument / parameter

A default parameter (also called an optional parameter or a default argument) is a function parameter that has a default value provided to it. If the user does not supply a value for this parameter, the default value will be used. If the user does supply a value for the default parameter, the user-supplied value is used instead of the default value.
e.g.

```
def printValues(int x, int y=10):
   print(x)
   print(y)

printValues(1)           // y will use default parameter of 10
printValues(3, 4)                  // y will use user-supplied value 4
```

Non Void Functions	Void Functions
Those functions which are returning values to the calling function	Those functions which are not returning values to the calling function.
Value return can be literal, variable , expression	We may use return but it will return none value to the function call

e.g.
def fun(a,b):
        c=a+b
        return c
x=int(input())
y= int(input())
z=fun(x,y)
print(z)
	e.g.
	def fun(a,b):
	        c=a+b
	        print(c)
	        return
	x=int(input())
	y= int(input())
	z=fun(x,y)
	print(z)

Arguments	**Parameters**
passed values in function call.	received values in function definition.
It can be of three types Literals Variables	It should be of variable(identifier) types.

Expressions	
def fun(a,b):	def fun(a,b): #parameters
c=a+b	c=a+b
print(c)	print(c)
x=2	x=2
y=4	y=4
fun(x,y) #variables	fun(x,y)
fun(5,6) #literals (values)	
fun(x+3,y+6) #expressions	

Mutable Data types	**Immutable Data types**
Object can be changed after it is created,	Object can't change its value in position after it is created.
Mutable is behaving like pass by reference	Immutable is behaving like pass by value
Mutable objects: list, dictionary	Immutable objects: int, float, complex, string, tuple
Everything in Python is an object ,and every objects in Python can be either mutable or immutable.	

Import Statement	**From Import Statement**
import all the modules from that package	only imports the required module as specified

r+	**w+**
Opens a file for reading and writing, placing the pointer at the beginning of the file.	Opens a file for writing and reading, overwrites the existing file if the file exists. If the file does not exist, creates a new file for writing and reading

r	**a**
Reading only	for appending
Sets file pointer at beginning of the file	Move file pointer at end of the file
This is the default mode.	Creates new file for writing,if not exist
e.g. f=open("abc.dat",'r')	e.g. f=open("abc.dat",'a')

TEXT FILE	**BINARY FILE**
A file whose contents can be viewed using a text editor is called a text file. (.txt)	A binary file stores the data in the same way as as stored in the memory.
A text file is simply a sequence of ASCII or Unicode characters.	Best way to store program information.
EOL (new line character i.e. enter) or internal translation occurs	No EOL or internal translation occurs(not converted into other form becoz it is converted into computer understandable form i.e. in binary format)
e.g. Python programs, contents written in text editors	e.g. exe files,mp3 file, image files, word documents

Relative Path	Absolute Path
The relative path is the path to some file with respect to current working directory	The absolute path is the full path to some place on your computer.
e.g. Relative path: "function.py" OR "..\function.py	For example: Absolute path: C:\Users\hp\Desktop\cs\function.py

seek()	tell()
takes the file pointer to the specified byte position	it gives current position within file

Syntax:
seek("no_of_bytes_to_move", "from_where")
 	
"from_where"- has 3 values
 	
from=
0 -means to move from the beginning of file. It is default also
1 means to move the pointer at the current position
2 means to move pointer at end of file
	Syntax
	fileobjectname.tell()
	Example:
	f.tell()

LIST	STRING
Lists are mutable	strings are immutable.
In consecutive locations, list stores the references of its elements.	In consecutive locations, strings store the individual characters
lists can store elements belonging to different types.	Strings store single type of elements-all characters
It is represented by []	It is represented by " " or ' '

e.g.
L=[1,2,3,4]
	e.g.
	s="hello"
	s1='world'

LIST	TUPLES
lists are mutable.	Tuples are immutable
List can grow or shrink	tuples cannot grow or shrink
For list []symbol is used	For tuples () symbol is used
e.g. L=[1,2,3,4]	e.g. T=(1,2,3,4)

LIST	DICTIONARY
lists are sequential collections(ordered)	dictionaries are non-sequential collections(unordered).
In LIST the values can be obtained using positions	But in dictionaries the values can be obtained using keys
this thing is not possible in list.	By changing the sequence of key we can shuffle the order of elements of dictionary
e.g. L=[1,2,3,4]	e.g. d={1:"hello",2:"world"}

Degree	Cardinality

It is the total number of attributes in the table.	It is the total number of tuples in the table

Candidate key	Primary key
A Candidate Key can be any column or a combination of columns that can qualify as unique key in database. There can be multiple Candidate Keys in one table.	A Primary Key is a column or a combination of columns that uniquely identify a record. Only one Candidate Key can be Primary Key.

Fetchone()	Fetchall()
fetchone() method returns one row or a single record at a time. It will return None if no more rows / records are available.	fetchall() fetches all the rows of a query result. An empty list is returned if there is no record to fetch the cursor.

Binary file	CSV file
▯ Extension is .dat ▯ Not human readable ▯ Stores data in the form of 0s and 1s e.g. .mp4, .mp3, .avi, .doc,.ppt,.jpg,.psd,.gif etc	▯ Extension is .csv ▯ Human readable ▯ Stores data like a text file Advantage of a csv file: ▯ It is human readable – can be opened in Excel and Notepad applications ▯ It is just like text file

COUNT(column_name)	COUNT(*)
used with Column_Name passed as argument and counts the number of non-NULL values in a column that is given as argument.	COUNT(*) returns the count of all rows in the table

Candidate key	Alternate key
All keys that have the properties to become a primary key are candidate keys.	The candidate keys that do not become primary keys are alternate keys.

Domain is a set of values from which an attribute can take value in each row. For example, roll no field can have only integer values and so its domain is a set of integer values.

A foreign key is used to set or represent a relationship between two relations (or tables) in a database. Its value is derived from the primary key attribute of another relation.

DDL	**DML**
Data definition language	Data manipulation language
Create	Insert
Alter	Update
Drop	Delete Select

ALTER	UPDATE
DDL command	DML command

It is used to add /delete/change name of a particular column	It is used to change/modify the value(s) in particular column(s)
alter table tablename add/modify/change column_name datatype(size)	update tablename set column_name=value where column_name=value

DROP	**DELETE**
DDL command	**DML command**
It is used to delete the table or database permanently	**It is used to delete a particular record(s) from the table**
drop table table_name	**delete from table_name [where condition]**

WHERE	**HAVING**
Where- Where clause is used to specify condition on single row.	having- It is used to mention condition in Group
Where clause is used mostly with Select, Update and Delete command/query	Having clause is used only with group by clause

ORDER BY	**GROUP BY**
It is used to arrange records in a particular order(asc/desc)	It is used to group together similar types of information
select */column_name from table_name order by column_name asc/desc	**Select */column_name from table_name group by column_name**