1. Write SQL statement select to display customer Full Name in one column, their City and Amount ofsales. We need data only for customers whose mother has brown eyes.

Sales

ID	CustomerID	CityID	Amount
1	3	2	500
2	1	1	10000
3	4	4	800
4	2	3	600
5	3	1	10000
6	1	1	630
7	3	1	960

Customer

Id	Gender	FirstName	LastName	EyeColor	IDNumber	MotherIDNumber	FatherIDNumber
1	Male	Peter	Cina	Blue	SK-156-232	NULL	NULL
2	Female	Adela	Cinova	Brown	SK-216-897	NULL	NULL
3	Female	Petra	Atkinson	Blue	SK-258-321	SK-216-897	SK-156-232
4	Male	Andrej	Nowak	Brown	SK-244-221	SK-411-897	SK-226-233
5	Female	Andrea	Atkinson	Green	SK-411-897	NULL	NULL
6	Male	Jozef	Jovanovic	Green	SK-226-233	NULL	NULL

Address

ID	Country	City
1	Slovakia	Presov
2	England	London
3	Slovakia	Bratislava
4	Slovakia	Trnava

Ans)

SELECT CONCAT(c.FirstName, ' ', c.LastName) AS FullName, a.City, s.Amount FROM Sales s INNER JOIN Customer c ON s.CustomerID = c.ID INNER JOIN Address a ON s.CityID = a.ID WHERE c.MotherIDNumber IN (SELECT IDNumber FROM Customer WHERE EyeColor = 'Brown');

2. Write SQL statement select to display First Name and Last Name of users which ordered 3 and morecourses. Use tables from below.

Course Table

ID	Name
1	Course A
2	Course B
3	Course C
4	Course D

User Table

ID	FirstName	LastName
1	Peter	Jovanovic
2	Jozef	Djordjevic
3	Milan	Atkinson
4	Maria	Armstrong
5	Slavomir	Cina
6	Robert	Varga
7	Peter	Nowak

Order Table

ID	UserID	CourseID
1	1	1
2	2	1
3	3	1
4	2	2
5	4	2
6	5	3
7	6	4
8	7	3
9	3	4
10	5	4
11	6	2
12	2	3

Ans)

```
SELECT u.FirstName, u.LastName
FROM User u
INNER JOIN Order o ON u.ID = o.UserID
GROUP BY u.ID
HAVING COUNT(DISTINCT
o.CourseID) >= 3;
```

3. What will be the result of the select below

```
SELECT SUM(p.Amount) AS Amount FROM
Payments p INNER JOIN Clients c ON
p.ClientId = c.Id INNER JOIN Address a ON
c.Id = a.ClientId WHERE c.Name LIKE '%iro
```

Clients

Id	Name	Age
1	Fero	14
2	Jozo	16
3	Miro	22
4	David	10
5	Vlado	35

Payments

Id	DueDate	Amount	ClientId
1	2016-07-08	100	1
2	2016-07-25	200	1
3	2016-09-08	300	2
4	2016-07-11	400	2
5	2016-11-12	500	3

Address

Id	Line 1	City	IsPrimary	ClientId
1	Fucikova 1	Bratislava	0	1
2	Jesenskeho 2	Trnava	1	1
3	Odborarska 3	Senec	0	1
4	Bottova 4	Malacky	0	3
5	Holleho 5	Topolcany	1	3

Ans)

Amount

800

PYTHON questions:

1. What is tuple in Python? What is the difference between list and tuple?

Tuples in Python:

A tuple in Python is an immutable sequence of values**.** This means that once a tuple is created, its elements cannot be changed. Tuples are often used to represent groups of related values that should not be modified.

Key characteristics of tuples:

- **Immutable:** Elements cannot be added, removed, or modified.
- **Ordered:** Elements maintain their order and can be accessed by index.
- **Heterogeneous:** Can contain elements of different data types.
- **Created using parentheses:** example_tuple = (1, 2.5, "hello", True)

Difference Between Lists and Tuples:

Feature	List	Tuple
Mutability	Elements can be changed	Elements cannot be changed
Creation	Created using square brackets []	Created using parentheses ()
Use Cases	Often used for dynamic data structures	Often used for fixed data structures or to prevent accidental modification
Performance	Generally, slightly slower due to the need for copying when modifying	Generally, slightly faster due to immutability

Example:

```
# List
my_list = [1, 2, 3]
my_list[0] = 10  # Valid, changes the first element

# Tuple
my_tuple = (1, 2, 3)
my_tuple[0] = 10  # Invalid, raises a TypeError
```

2. What are the rules for a local and global variable in Python?

Local Variables:

- Defined within a function.
- Scope is limited to the function where they are defined.
- Cannot be accessed outside the function.
- Any assignment within a function creates a local variable, unless explicitly declared as global.

Global Variables:

- Defined outside of any function.
- Can be accessed from anywhere in the program, including inside functions.
- To modify a global variable within a function, you must explicitly declare it as global using the global keyword.

Rules:

- If a variable is referenced inside a function without being assigned a value, Python assumes it's a global variable.
- If a variable is assigned a value within a function, it's assumed to be a local variable unless explicitly declared as global.
- To modify a global variable within a function, use the global keyword before assigning a value to it.

Example:

```
x = 10  # Global variable

def my_function():
    y = 20  # Local variable
    global x
    x = 30  # Modifying global variable

my_function()

print(x)  # Output: 30
print(y)  # Error: y is not defined (local variable)
```

3. What is Python's parameter passing mechanism? Name it and explain it.

Python's parameter passing mechanism is called "pass by reference."

While this might sound like Python passes actual references to objects, it's more accurate to say that it passes object references. When a function is called, the arguments passed to it are actually references to the objects being passed. This means that any changes made to the objects within the function will be reflected outside the function as well.

Here's a breakdown:

1. **Object Creation:** When you create an object in Python, a reference to that object is stored in a variable.
2. **Function Call:** When you pass an object as an argument to a function, you're essentially passing a copy of the reference to that object.
3. **Modifications:** If the function modifies the object using its reference, the original object will also be changed.

Example:

```
def modify_list(my_list):
    my_list.append(4)

my_list = [1, 2, 3]
modify_list(my_list)
print(my_list)  # Output: [1, 2, 3, 4]
```

In this example, the modify_list function takes a list as an argument. When the function is called, a copy of the reference to the my_list object is passed. Inside the function, the `append` method is used to add an element to the list, modifying the original object.

4. Write a method to open a text file and display its content?

```
def read_file(file_path):
    with open(file_path, 'r') as file:
        print(file.read())
```

5. You have two lists: strList = ["Vishesh", "For", "Python"] and valList = [1, 2] for the first two tasks and one list valList = [1, 2, 3] for third task. Write the sintax so you will get these results:

1) {'key2': ['Vishesh', 'For', 'Python'], 'key1': [1, 2]}
2) {'key1': [1, 2, ['vishesh', 'For', 'python']]}
3) {'1': [1, 2], '3': [3, 4], '2': [2, 3]} # Creating a dictionary of lists using list comprehension.

Ans)

1)

```
strList = ["Vishesh", "For", "Python"]
valList = [1, 2]
result = {'key2': strList, 'key1': valList}
```

2)

```
strList = ["Vishesh", "For", "Python"]
valList = [1, 2]
result = {'key1': valList + [strList]}
```

3)

```
valList = [1, 2, 3]
result = {str(val): [val, val + 1] for val in valList}
```