1.What is the relationship between def statements and lambda expressions ?

Ans: Lambda are operators can have any number of arguments, but it can have only one expression. It cannot contain any statements and it returns a function object which can be assigned to any variable. They can be used in the block they were created.

Functions help break our program into smaller and modular chunks. As our program grows larger and larger, functions make it more organised and manageable. They can be called and used anywhere we want.

2. What is the benefit of lambda?

Ans: Lambda functions reduce the number of lines of code when compared to normal python function defined using def keyword

3. Compare and contrast map, filter, and reduce.

Ans: The map() function iterates through all items in the given iterable and executes the function we passed as an argument on each of them.

filter() forms a new list that contains only elements that satisfy a certain condition, i.e. the function we passed returns True

reduce() works differently than map() and filter(). It does not return a new list based on the function and iterable we've passed. Instead, it returns a single value.

4. What are function annotations, and how are they used?

Ans :Function annotations are arbitrary python expressions that are associated with various part of functions. These expressions are evaluated at compile time and have no life in python's runtime environment

5. What are recursive functions, and how are they used?

Ans: A recursive function is a function that calls itself during its execution. The process may repeat several times, outputting the result and the end of each iteration. A recursive function calls itself, the memory for a called function is allocated on top of memory allocated to calling function and different copy of local variables is created for each function call

6. What are some general design guidelines for coding functions?

Ans: 1. Limited use of globals

2.Standard headers for different modules

3.Naming conventions for local variables, global variables, constants and functions

4. Indentation

5.Error return values and exception handling conventions

6.Avoid using a coding style that is too difficult to understand

7.Avoid using an identifier for multiple purposes

8.Code should be well documented

9.Length of functions should not be very large

10.Try not to use GOTO statement

7. Name three or more ways that functions can communicate results to a caller.

Ans: Using Positional Notation

Positional notation is the traditional mechanism for passing arguments to functions

e.g, SELECT concat\_lower\_or\_upper('Hello', 'World', true);

Using Named Notation

In named notation, each argument's name is specified using := to separate it from the argument expression.

e.g,SELECT concat\_lower\_or\_upper(a := 'Hello', b := 'World')

Using Mixed Notation

The mixed notation combines positional and named notation.

e.g, SELECT concat\_lower\_or\_upper('Hello', 'World', uppercase := true)