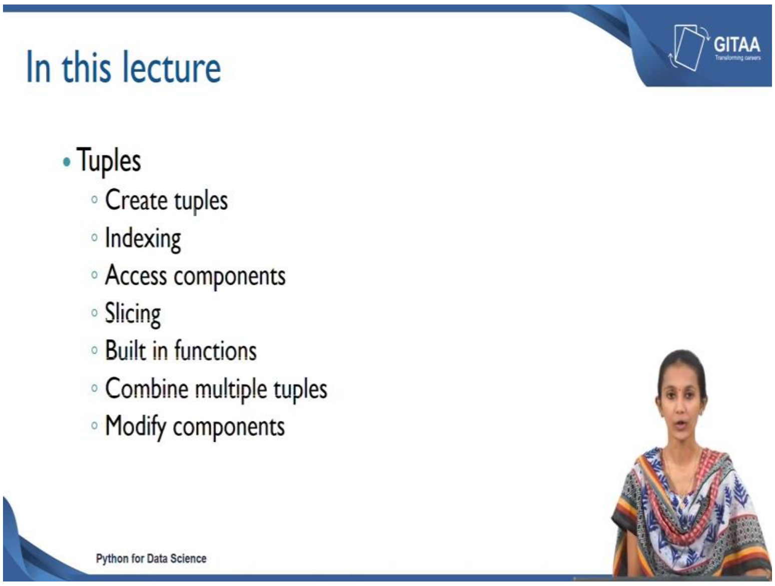
Python for Data Science
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Lecture – 09
Tuples

Welcome to the lecture.

(Refer Slide Time: 00:19)

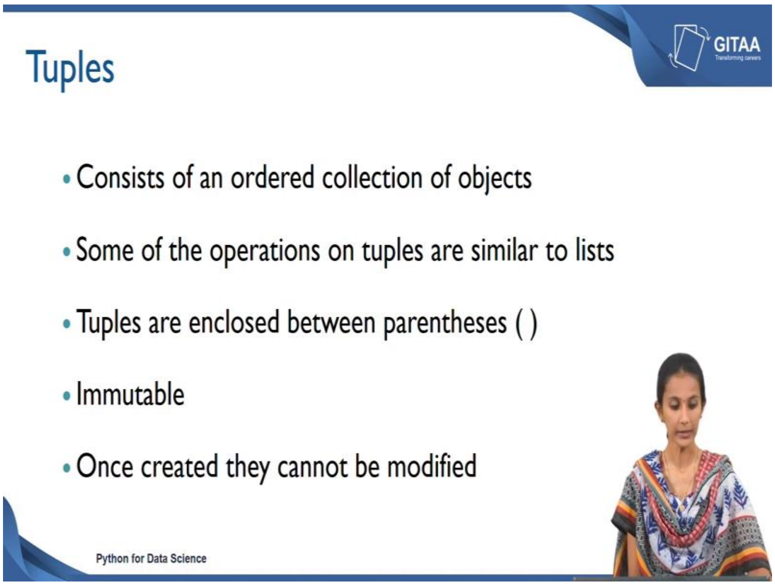


In this lecture

- Tuples
  - Create tuples
  - Indexing
  - Access components
  - Slicing
  - Built in functions
  - Combine multiple tuples
  - Modify components

In this lecture we will see, what is mean by Tuples, how to create tuples, indexing, how to access the components of the tuple. We will also see slicing and some of the built in functions available in tuples. We will also see how to combine multiple tuples that is two or more tuples, we will also see how to modify components of the tuples.

(Refer Slide Time: 00:45)



Tuples

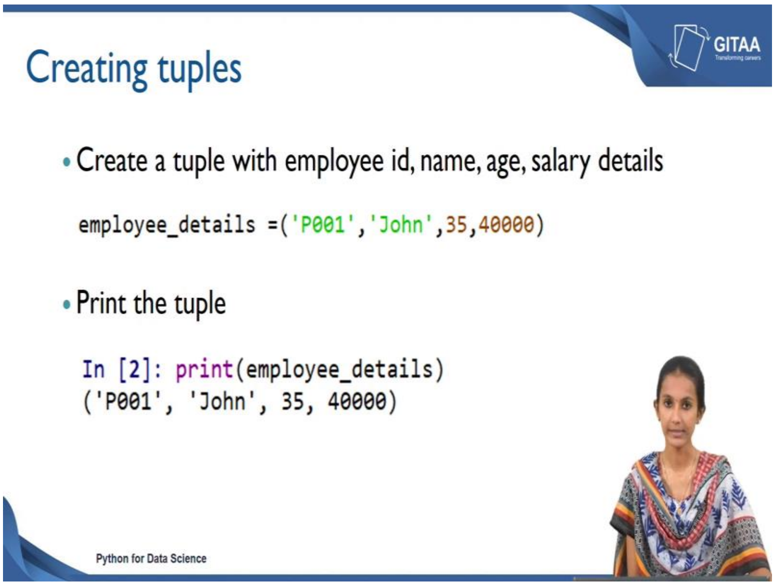
- Consists of an ordered collection of objects
- Some of the operations on tuples are similar to lists
- Tuples are enclosed between parentheses ( )
- Immutable
- Once created they cannot be modified

Python for Data Science

Let us get started, we will see what is mean by tuples. Tuples is one of the data structure available in python. It consists of an ordered collection of objects or elements. Some of the operations on tuples which are similar to list as well so, you can access the components of the lists. So, if you wanted to do slicing operations as well so, you can do it on the tuples.

So, the tuples values are in separated by the commas and enclosed between the parentheses. The tuples are immutable. So, if you have created a tuple, you will not be able to modify the tuple later.

(Refer Slide Time: 01:19)



Creating tuples

- Create a tuple with employee id, name, age, salary details

```
employee_details =('P001','John',35,40000)
```

- Print the tuple

```
In [2]: print(employee_details)
('P001', 'John', 35, 40000)
```

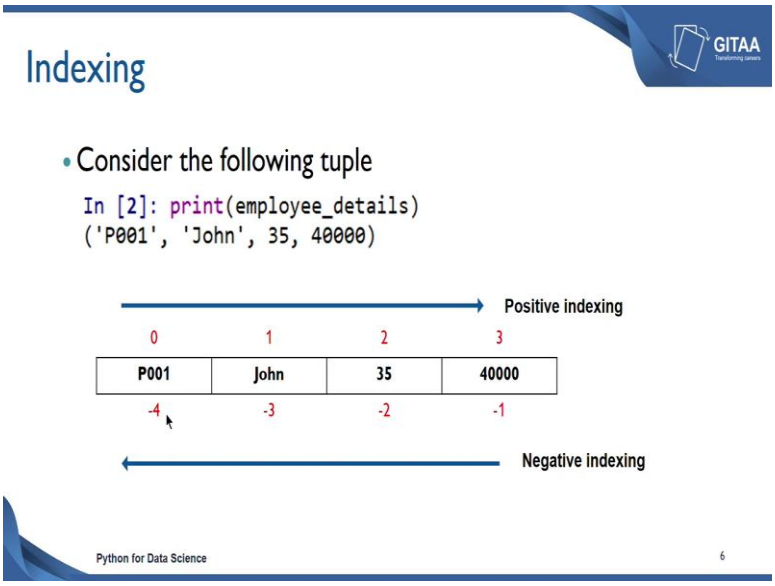
Python for Data Science

So, let us see how to create a tuple. So, I want to told you create a tuple with employee id, employee name, age and salary details. So, this basically forms our employee_details. So, the tuple has to be created using a parenthesis separated by commas.

So, basically what I will do is I will create a tuple employee_details which contains the id P001 which is the string is basically an id, John is an employee name, 35 is a numeric data type which is corresponds to age, 40000 which corresponds to the salary details. So, I wanted to create a tuples using these details.

So, let us say if I wanted to print the tuple employee_details. So, you have to give the print(employee_details). So, whatever you have created as a tuples so, it will be displayed with those details. So, we will have id we will have a employee name, will have a age and as well as the salary details.

(Refer Slide Time: 02:27)



Indexing

- Consider the following tuple

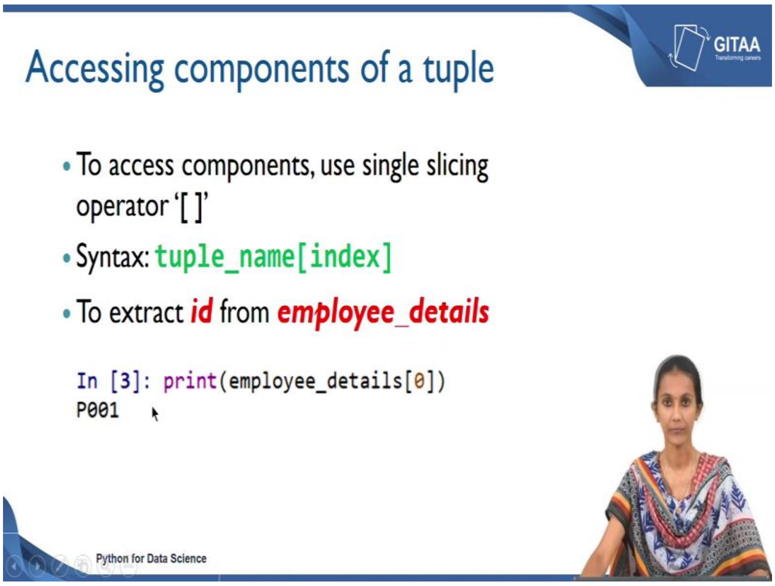
```
In [2]: print(employee_details)
('P001', 'John', 35, 40000)
```

Next we will look at the indexing in python. Since a tuple is an ordered collection of elements so, each items can be accessed individually. So, if you wanted to slice out the some of the elements as well. So, you can do it in the tuple.

So, we have the employee_details which has a tuple with id, employee name, age and salary. So, let us see the index number. So, we have two types of indexing. So, that is one is positive indexing and the other one is negative indexing. So, the positive indexing, it always starts from left to right and negative indexing it starts from right to left for the positive indexing the index number, it always starts from 0.

So, the id as corresponds to the index number 0, employee name John; it corresponds to the index number 1, age which is 35 corresponds to the index number 2, salary details which corresponds to the index number 3. So, this is the positive indexing. Next negative indexing, it always starts from right to left so, our 40000 which is a salary detail, its starts with the index number of -1, age it corresponds to the index number of -2, employee name it corresponds to the index number of -3, id will corresponds to the index number of -4. So, this is called as a negative indexing.

(Refer Slide Time: 04:03)



Accessing components of a tuple

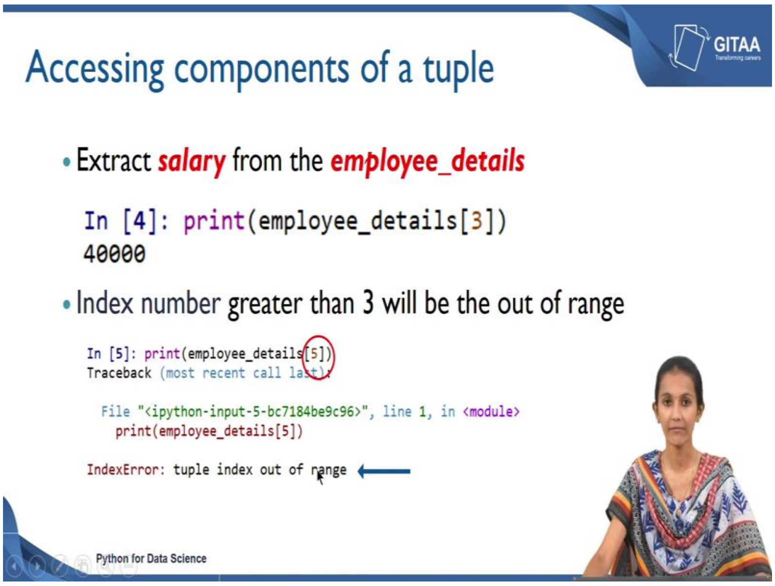
- To access components, use single slicing operator '[ ]'
- Syntax: tuple_name[index]
- To extract *id* from *employee_details*

```
In [3]: print(employee_details[0])
P001
```

Let us say if I wanted to access some of the components of the tuple, let us see how to do that to access the components. We need to use the single slicing operator which is a square brackets. So, basically the syntax is you have to give the tuple name followed by the in the single slicing operator, we have to specify the index number.

Let us say if I wanted to extract id from the employee_details already, we saw that for id, it corresponds to the index number of 0. So, for positive indexing for negative indexing, it corresponds to the index number of -4. So, I wanted to extract id from the employee_details. So, employee_details is a tuple. So, if we give employee_details followed by the index number so, id corresponds to the index number of 0 right. So, if you give that you will be able to access the id from the tuple employee_details.

(Refer Slide Time: 05:07)



Accessing components of a tuple

- Extract ***salary*** from the ***employee_details***

```
In [4]: print(employee_details[3])
40000
```

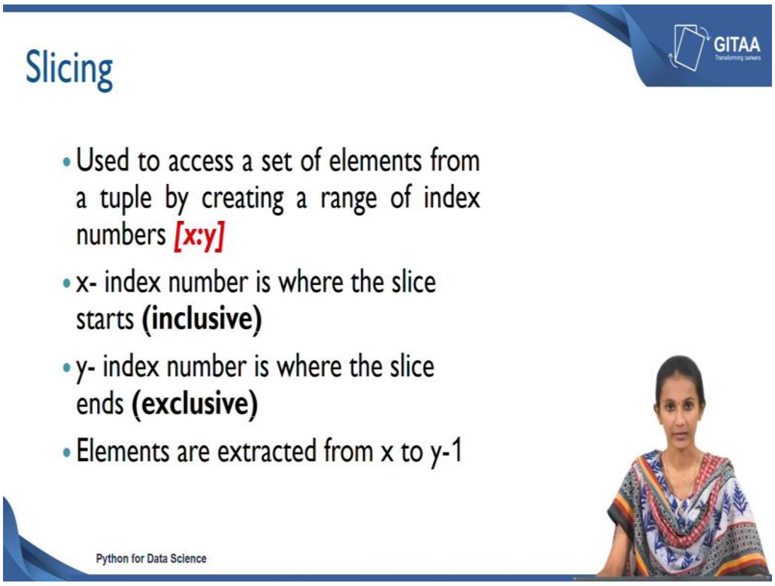
- Index number greater than 3 will be the out of range

```
In [5]: print(employee_details[5])
Traceback (most recent call last):

  File "<ipython-input-5-bc7184be9c96>", line 1, in <module>
    print(employee_details[5])

IndexError: tuple index out of range
```

Let us say if I wanted to extract salary from the employee_details. So, salary corresponds to the index number of 3 right. So, if we give the tuple name followed by the index number, you will be able to extract salary from the employee_details tuples which basically gives you the value of 40000.

So, you can also use the negative indexing value as well. We have the index number starting from 0 till 3. So, if I give index number greater than 3, it will be the out of range. So, let us say if I wanted to access the 5th index number print employee_details 5. So, it will throws you an error of tuple index out of range which means so, we have the index number its starting from 0 till 3. So, you do not have the value of index number 5 which says tuple index out of range. Next we look at slicing.

(Refer Slide Time: 06:07)



Slicing

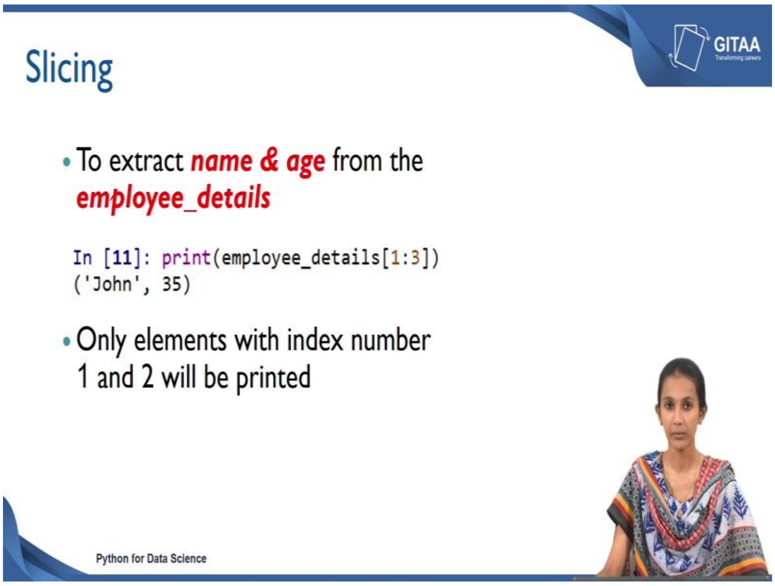
- Used to access a set of elements from a tuple by creating a range of index numbers *[x:y]*
- x- index number is where the slice starts **(inclusive)**
- y- index number is where the slice ends **(exclusive)**
- Elements are extracted from x to y-1

GITAA
Transforming careers
Python for Data Science

So, slicing is basically used to access the set of elements. If you have wanted to access the multiple elements from the tuple so, you will use the range of index numbers which is x is to y let us see what is x and what is y. So, x is basically the index number where the slice should just starts. So, this is inclusive. So, this number will be included in the tuple. So, y it is the index number where the slices should end.

So, if you give the value of 2 so, that index number will be excluded from the tuple. So, y is the index number where the slice should ends. So, if you give the index number of 2, tuple will be getting output of the range till index number of 1. So, the elements are extracted from x to y-1. So, the x index number will be included for y it is exclusive. So, it will be y -1.

(Refer Slide Time: 07:09)



Slicing

- To extract ***name & age*** from the ***employee_details***

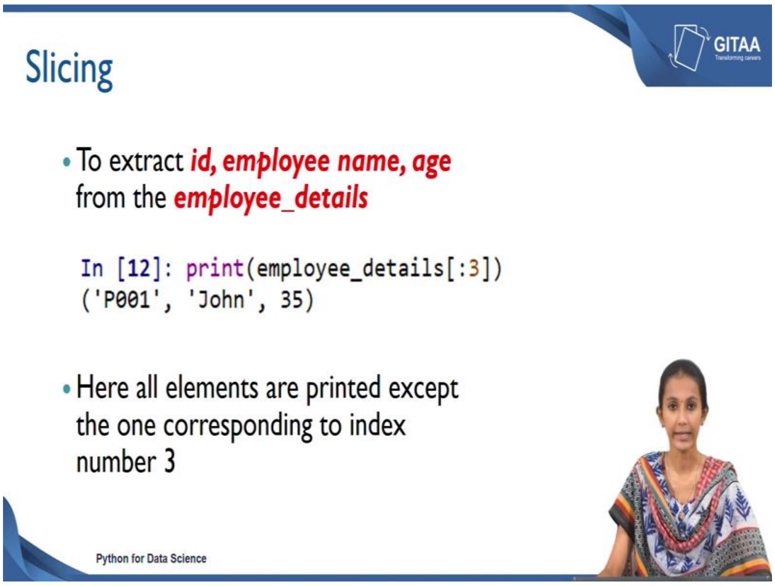
```
In [11]: print(employee_details[1:3])
('John', 35)
```

- Only elements with index number 1 and 2 will be printed

Let us say if I wanted to extract two elements from the employee_details tuple. So, I wanted to extract name as well as age. So, if you get details the name corresponds to the index number of 1 ; employee name corresponds to the index number of 1, age corresponds to the index number of 2, salary details corresponds to the index number of 3.

So, if you give 1:3 what it does is, 1 will be included. So, the index number will be included 3 will not be included. So, it will be 3-1 so, it will be 2. So, we will be getting an output of John and 35 so, which corresponds to the employee name and age. So, only elements with the index number of 1 and 2 will be printed 3 will not be printed in this case.

(Refer Slide Time: 08:05)



Slicing

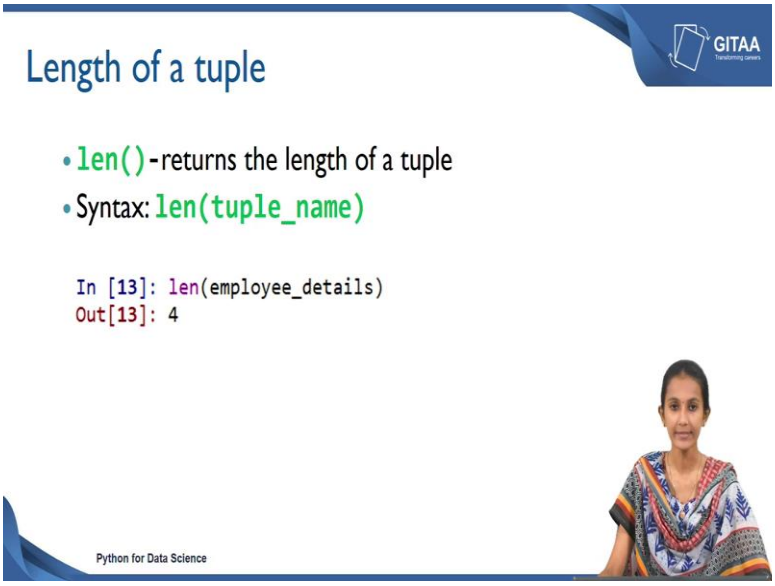
- To extract ***id, employee name, age*** from the ***employee_details***

```
In [12]: print(employee_details[:3])
('P001', 'John', 35)
```

- Here all elements are printed except the one corresponding to index number 3

Let us say if I wanted to extract id, employee name, age from the employee_details. So, we have four elements right a id, employee name, age and salary details, but I wanted to extract only the 3 elements from the employee_details. So, if we give print(employee_details[:3]) so, it will be able to print id, the employer name and age values. So, the id will be having P001 John and the age value which is 35. So, the all the elements will be printed except the corresponding to index number 3. So, it will be 3 -1 so, the elements will be printed till the index number 2.

(Refer Slide Time: 08:57)



Length of a tuple

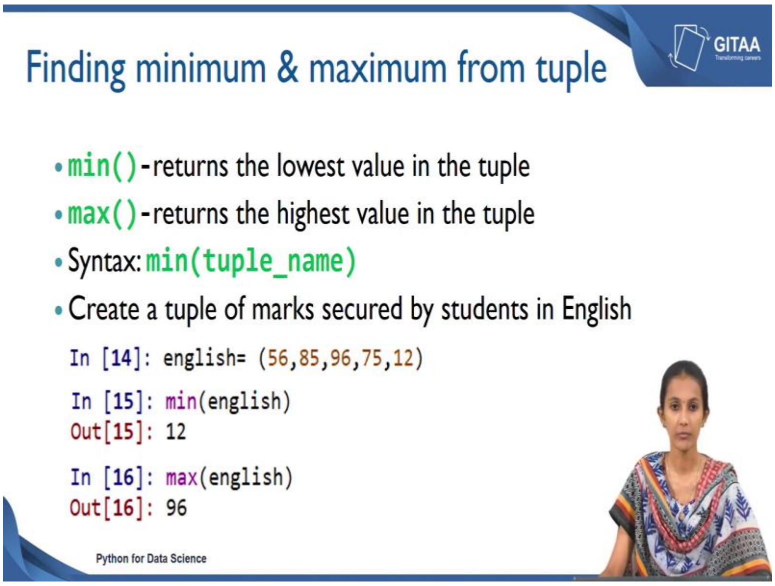
- `len()` - returns the length of a tuple
- Syntax: `len(tuple_name)`

```
In [13]: len(employee_details)
Out[13]: 4
```

So, let us says if I wanted to calculate the length of a tuple. So, if you have more elements in the tuple when you are created so, let us say I wanted to check the length of the tuple. So, for command is len, it basically returns the length of the tuples, which means a number of elements in the tuple. The syntax is len followed by the tuple name which we have declared earlier.

So, in our case our tuple name is employee_details. So, if you give length of employee_details so, it returns you an value of 4 which means so, you have four elements in the tuples. So, basically we have id, employee name, age and the salary details.

(Refer Slide Time: 09:45)



Finding minimum & maximum from tuple

- min()-returns the lowest value in the tuple
- max()-returns the highest value in the tuple
- Syntax: min(tuple_name)
- Create a tuple of marks secured by students in English

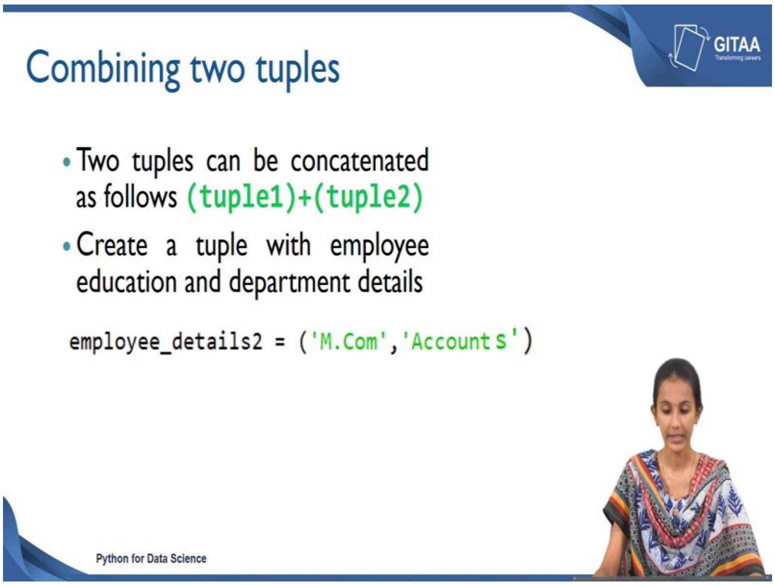
```
In [14]: english= (56,85,96,75,12)

In [15]: min(english)
Out[15]: 12

In [16]: max(english)
Out[16]: 96
```

Next we will let us see how to calculate the minimum and maximum from the tuple. So, if you have a quantitative data with integer or numeric value. So, if you wanted to calculate the minimum and maximum so, you can use the built in functions which are available in python.

So, min() is a built in function. So, if you wanted to calculate the lowest value in the tuples so, you can use the min function. So, if I wanted to calculate the highest value in the tuple so, we can use the max function. So, syntax for min is min followed by the tuple name. So, now, I will create a tuple of marks secured by students in English subject.

So, now I am creating a tuple called English which has sets of values, 56, 85, 96, 75 and 12. So, while creating a tuple the element should be separated by commas. So, I wanted to calculate the minimum value for this marks secured by the students in English. So, if you find out so, the minimum value will be 12. So, we will use the built in function as well. So, if you give min of English, you will be getting an output of 12; max of English returns you an values of 96.

(Refer Slide Time: 11:13)



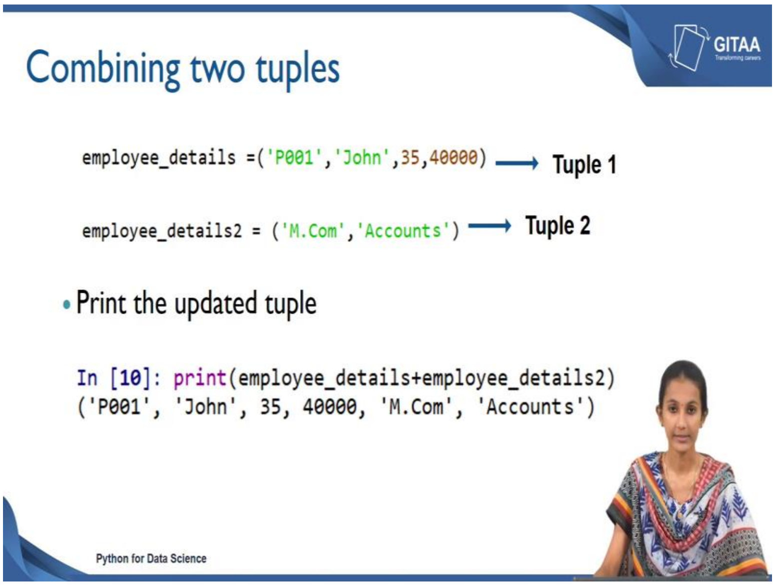
Combining two tuples

- Two tuples can be concatenated as follows (tuple1)+(tuple2)
- Create a tuple with employee education and department details

```
employee_details2 = ('M.Com','Accounts')
```

Let us look at how to combine the 2 tuples or more tuples. So, 2 tuples can be concatenated by using the operations which are available in python. So, if we have tuple1+tuple2, we can combine the 2 tuples. So, we can also combined the more tuples as well. So, we already created a tuple with a employee_details. Now, I am going to create an another tuple with employee education and department details. So, we already had employee_details which as a id, age, employee name and salary details. So, now, I am creating an employee_details 2 which as the education, M.com and the department details he belongs which is Accounts.

(Refer Slide Time: 12:05)



Combining two tuples

```
employee_details =('P001','John',35,40000)
```

```
employee_details2 = ('M.Com','Accounts')
```

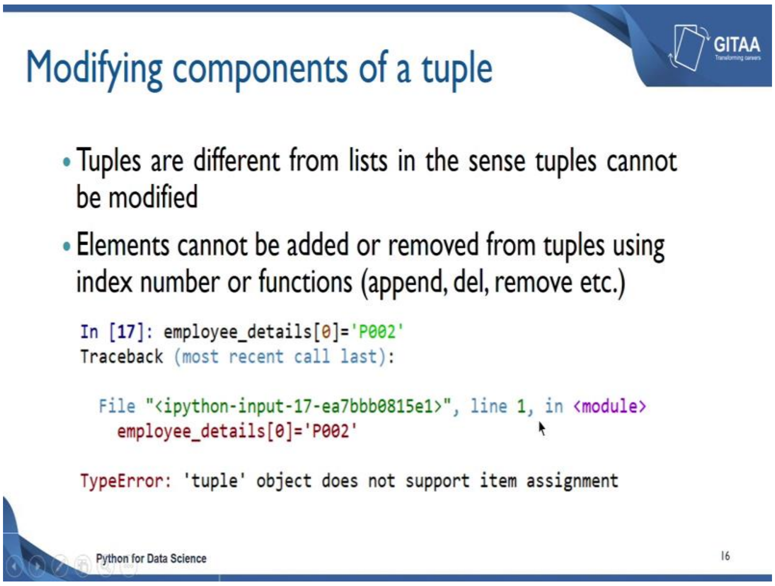
- Print the updated tuple

```
In [10]: print(employee_details+employee_details2)
('P001', 'John', 35, 40000, 'M.Com', 'Accounts')
```

Now, we will see how to combine the 2 tuples. This is our tuple1 and we have created employee_details 2 which is our tuple2. So, I am going to combined the tuple1 and tuple2 together. So, we can also store it in a some other variable name or else if you wanted to print it as well you can print the 2 tuples by using the plus symbol. So, I wanted to print the updated tuple.

So, if you give print(employee_details + employee_details2). So, you can see the combined 2 tuples; those now we will have id, employee name, age, salary details and as well as a education and as well as a department which he belongs to.

(Refer Slide Time: 12:55)



Modifying components of a tuple

- Tuples are different from lists in the sense tuples cannot be modified
- Elements cannot be added or removed from tuples using index number or functions (append, del, remove etc.)

```
In [17]: employee_details[0]='P002'
Traceback (most recent call last):

  File "<ipython-input-17-ea7bbb0815e1>", line 1, in <module>
    employee_details[0]='P002'

TypeError: 'tuple' object does not support item assignment
```

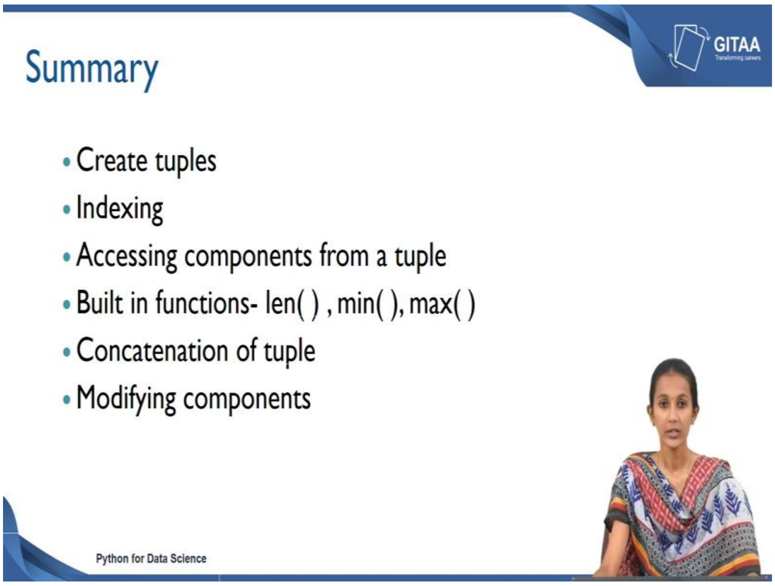
Python for Data Science

16

Next we will see how to modify the components of a tuple. So, the tuples are different from list in the sense, he will not be able to modified the tuples once created. So, in this we saw how to add or remove the elements from the lists right, but in the tuple we will not be able to add or remove from the tuples using as well as a index numbers or as well as the built in functions. So, append del remove those functions will not be working in this tuple.

So, in this case, I wanted to change the id. So, already we had corresponding to the index number 0 we had P001. But now I wanted to change the id value. So, now, I am replacing with P002 corresponds to the index number of 0, but it throws me an error of type error tuple object does not support item assignment. Once the tuple has been created will not be able to modify which means you will not be able to add or remove elements from the tuple.

(Refer Slide Time: 14:05)



Summary

- Create tuples
- Indexing
- Accessing components from a tuple
- Built in functions- len( ) , min( ), max( )
- Concatenation of tuple
- Modifying components

So, let us summarize. So, we saw how to create tuples values inside the tuples are separated by commas and enclosed by the parenthesis. We saw the indexing. So, there are 2 types of indexing one is positive indexing and the other one is negative indexing. So, positive indexing it always starts from 0 to n-1. So, the negative indexing, it starts from the rightmost to the leftmost. We also saw how to access the components from a tuple so, you need to specify the index numbers in the tuples. So, if I wanted to extract age from the tuples, you have to specify the index number.

We also saw how to extract 2 or more elements from the tuple using the slicing. So, let us say if I wanted to extract 2 elements from the tuple, you need to specify the x and y values. So, x is the index number will be included in the tuple, y the index number will not included. So, it will be elements will be extracted from x to y -1 we also saw some of the built in functions which are available in python which is the length its written the length of the tuple which means the number of elements in the tuple. We also saw the minimum function and maximum function. So, these can be used for the quantitative data.

We also saw concatenation of a tuple. So, how to combine the 2 tuples? So, you can also use it +and you are going to if you wanted to multiple the tuples as well you can use the multiple function as well. So, we also saw concatenation of a tuple how to combine 2 or more tuples. So, you can use the + operator. We also saw the modifying components in the tuple, we will not be able to modify the components using the index number or as well as the built in functions as well. The major advantage is so, once you have created, the tuples will not be able to modify.