/\*==============================================================\*/

/\* **COLUMN GRAPH** \*/

/\*==============================================================\*/

|  |
| --- |
| import cx\_Oracle |
|  | import plotly.offline as py |
|  | import plotly.graph\_objs as go |
|  |  |
|  |  |
|  | connection = cx\_Oracle.connect("project", "project", "xe") |
|  |  |
|  | cursor = connection.cursor() |
|  |  |
|  | cursor.execute(""" |
|  | select comment\_number, comment\_text |
|  | from exercise |
|  | """) |
|  |  |
|  | comment\_number = [] |
|  | comment\_text = [] |
|  |  |
|  |  |
|  | for row in cursor: |
|  | print("comment number: ",row[0]," and comment: ",row[1]) |
|  | comment\_number += [row[0]] |
|  | comment\_text += [row[1]] |
|  |  |
|  | data = [go.Bar( |
|  | x= comment\_number, |
|  | y= comment\_text |
|  | )] |
|  |  |
|  | layout = go.Layout( |
|  | title='comment and comment\_nubmer', |
|  | xaxis=dict( |
|  | title='comment', |
|  | titlefont=dict( |
|  | family='Courier New, monospace', |
|  | size=18, |
|  | color='#7f7f7f' |
|  | ) |
|  | ), |
|  | yaxis=dict( |
|  | title='comment\_nubmer', |
|  | rangemode='nonnegative', |
|  | autorange=True, |
|  | titlefont=dict( |
|  | family='Courier New, monospace', |
|  | size=18, |
|  | color='#7f7f7f' |
|  | ) |
|  | ) |
|  | ) |
|  | fig = go.Figure(data=data, layout=layout) |
|  |  |
|  | lection\_comment =py.plot(fig) |

/\*==============================================================\*/

/\* **Dynamic graph** \*/

/\*==============================================================\*/

|  |
| --- |
| import cx\_Oracle |
|  | import plotly.offline as py |
|  | import plotly.graph\_objs as go |
|  | connection = cx\_Oracle.connect("project", "project", "xe") |
|  |  |
|  | cursor = connection.cursor() |
|  |  |
|  | cursor.execute(""" |
|  | select lection.lection\_id, lection\_them.number |
|  | from lection join lection\_them on lection.lection\_id = lection\_them.lection\_id |
|  | """) |
|  |  |
|  | datetime = [] |
|  | them = [] |
|  |  |
|  | lection\_id = int(input('Input lection\_id: ')) |
|  |  |
|  | for row in cursor: |
|  | if row[0] ==lection\_id: |
|  | them += [row[1]] |
|  | datetime += [row[2]] |
|  | print("Date: ",row[1],"them",row[2]) |
|  | else: |
|  | continue |
|  |  |
|  |  |
|  | them\_dates = go.Scatter( |
|  | x=datetime, |
|  | y=them, |
|  | mode='lines+markers' |
|  | ) |
|  | data = [them\_dates] |
|  | py.plot(data) |