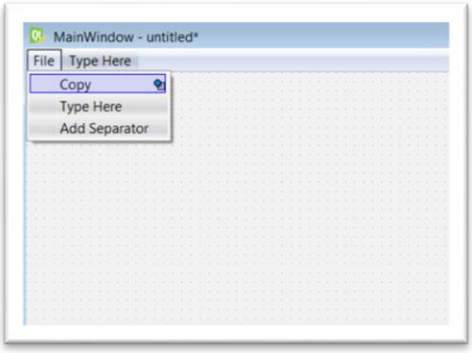
Question 1 (Advanced Widgets)

1. What are widgets and why are they usefull in programming? (2)
2. Name any 3 widgets that you have used in programming (3)
3. Explain how you would add a LCD widget and how you can use this to show the system time (5)
4. Explain how you would add a calendar widget and how you can use this to display the date selected in a date edit widget.
5. Why is a combox box usefull in applications? (1)
6. Explain how can you use a combo box in a GUI (3)

Question 2 (Menus and Toolbars)

Use the diagram below to unswer the questions that follow.



Questions

1) Explain how would you create a Menu bar with File and under File three placeholders with Edit, copy and paste (2)
2) Explain how you would you create a shortcut key for the copy and paste submenus (4)
3) How is a menu bar different from a toolbar (2)
4) What is a placeholder? (1)
5) How would you add a separator in a menu to create a nested menu? (2)
6) Explain how does an Action Editor works (4)

Creating a toolbar

Questions

1) What is a toolbar useful for (1)
2) Explain briefly how you would create an action with an icon

DOCK WIDGET

Questions

1) What is a docked widget and why is it useful?
2) Name 4 areas that you can place a dock widget
3) What property should you enable to make the dock area movable

TAB WIDGET

Questions

1. What is the purpose of a tab widget
2. How do you convert a tab to a widget
3. How do you change the style, gradient and colour of a tabbed document?

Other Questions on Menus

1) Give 4 reasons why it is important to use Menus in an application (3)
2) Give the necessary steps on how to add a menu to an item (2)
3) Explain how to attach code to menu items? Give an example by refering to an application. 3
4) Explain the different ways that can be used to trigger the same operation. (3)
5) Discuss 3 reasons why menus are required in the development of a database application (2)

Question 3 (Multiple Document Interface)

1. Describe the characteristics of a Multiple Document Interface (MDI) – (5)
2. Discuss 3 Reasons/benefits for the use of multiple forms in an application AND give motivation with reference to the development of the project application. (3)
3. Identify 2 modes that you can use in a database project.
4. Use the code below to fill in the code where specified.

```
import sys
from mdidemo import *

class MyForm(QtGui.QMainWindow):
   def __init__(self, parent=None):
      QtGui.QWidget.__init__(self, parent)
      self.ui = Ui_MainWindow()
      self.ui.setupUi(self)
      self.ui.mdiArea.addSubWindow(self.ui.Main)
      self.ui.mdiArea.addSubWindow(self.ui.Tab1)
      self.ui.mdiArea.addSubWindow(self.ui.Tab2)
      self.ui.mdiArea.addSubWindow(self.ui.Tab3)
      self.ui.Main.setWindowTitle("Main")
      QtCore.QObject.connect(self.ui.button1, QtCore.SIGNAL('clicked()'), self.displayNext)
      QtCore.QObject.connect(self.ui.ShowPrevious, QtCore.SIGNAL('clicked()'),
self.displayprevious)
      QtCore.QObject.connect(self.ui.closeAll, QtCore.SIGNAL('clicked()'), self.closeAll)
      QtCore.QObject.connect(self.ui.cascadeButton, QtCore.SIGNAL('clicked()'),
self.cascadeArrange)
      QtCore.QObject.connect(self.ui.tileButton, QtCore.SIGNAL('clicked()'), self.tileArrange)
      QtCore.QObject.connect(self.ui.SubWindowViewButton, QtCore.SIGNAL('clicked()'),
                  self.SubWindowView)
      QtCore.QObject.connect(self.ui.TabbedViewButton, QtCore.SIGNAL('clicked()'),
self.TabbedView)
      QtCore.QObject.connect(self.ui.actionFirst_Window, QtCore.SIGNAL('triggered()'),
                  self.displayNext)
      QtCore.QObject.connect(self.ui.actionSecond_Window, QtCore.SIGNAL('triggered()'),
                  self.displayprevious)

   def displayNext(self):
    self.ui.Court2.setFocus()


   def displayprevious(self):
    self.ui.mdiArea.activatePreviousSubWindow()
```

```
        def closeAll(self):
            Enter code to close all subwindows

        def cascadeArrange(self):
            Enter code to cascade all subwindows

        def tileArrange(self):
            Enter Code to arrage windows in a tile fashion

        def SubWindowView(self):
            Enter code to set view to Window View

        def TabbedView(self):
            Enter code to set view to tabbed view

    if __name__ == "__main__":
        app = QtGui.QApplication(sys.argv)
        myapp = MyForm()
        myapp.show()
        sys.exit(app.exec_())
```

Question 4 (Database handeling)

1. What are the benefits of storing information in a databace? (3)
2. What is the difference between data and information?
3. What are the benefits of using MySql (6)
4. Give the commands you would use to create a database called shopping using MySql.
    a. What is the command used to create a database called shopping (1)
    b. How would you display any tables in the shopping database (1)
    c. Suppose you have created a table called products. How would you disply the contents of products. (1)
5. Use the code below too
    a. Create a new table called cars with name, surname, date, address and age into the database. Provide the single line of code only
    b. Add the following values into the database. Mike, Praisley. 23-12-1995, 23 First road, 67. Provide the single line of code only

```
#InsertRow.py
import sys
import mysql.connector
conn = mysql.connector.connect(host="localhost",user="root",password="nowin",database="booking")
cursor = conn.cursor()
cursor.execute(Insert code here to add the items)")

print ("One Row Added")

cursor.close()
conn.commit()
conn.close
```

6. Use the program below to comment on the code given. You do not need to type out the entire code, only comment. (8)

```
import sys
from assignment2 import *
from PyQt4 import QtSql, QtGui

def createConnection():
    db = QtSql.QSqlDatabase.addDatabase('QMYSQL')
    db.setHostName('localhost')
    db.setDatabaseName('booking')
    db.setUserName('root')
    db.setPassword('nowin')
    db.open()
    print (db.lastError().text())
    return True

class MyForm(QtGui.QMainWindow):
    def __init__(self, parent=None):
        QtGui.QWidget.__init__(self, parent)
        self.ui = Ui_MainWindow()
        self.ui.setupUi(self)
    #Comment here
        self.model = QtSql.QSqlTableModel(self)
#Comment here
        self.model.setTable("players2")
#Comment here
        self.model.setEditStrategy(QtSql.QSqlTableModel.OnManualSubmit)
#Comment here
        self.model.select()
#Comment here
        self.ui.tableView.setModel(self.model)
        #connects the button UpdateButton to the function UpdateRecords.
        QtCore.QObject.connect(self.ui.UpdateButton, QtCore.SIGNAL('clicked()'),
                    self.UpdateRecords)
        QtCore.QObject.connect(self.ui.CancelButton, QtCore.SIGNAL('clicked()'),
                    self.CancelChanges)
        QtCore.QObject.connect(self.ui. DeleteButton, QtCore.SIGNAL('clicked()'),
                    self.DeleteRecords)

    def UpdateRecords(self):
    #submits the current edited row
     self.model.submitAll()

    def CancelChanges(self):
    #cancels and changes
     self.model.revertAll()

    def DeleteRecords(self):
    #deletes a row
     self.model.removeRow(self.ui.tableView.currentIndex().row())
     self.model.submitAll()

if __name__ == "__main__":
    app = QtGui.QApplication(sys.argv)
    if not createConnection():
        sys.exit(1)
    myapp = MyForm()
    myapp.show()
    sys.exit(app.exec_())
```