MySQL for Python

```
                    ## CONECTANDO ##
# Importar lib mysqldb
import  MySQLdb

# Conectando ao banco de dados
db =
MySQLdb.connect("localhost", "usuario", "senha", "nomed
b")

# Preparar objeto cursor usando método cursor()
cursor = db.cursor()

                  ## INSERINDO TABELAS ##
# Eliminar tabela se existir
cursor.execute("drop table if exists TABELA")

# Cria uma tabela (exemplo)
sql = """create table TABELA(
         nome char(30) not null,
         idade int,
         salario float)"""
cursor.execute(sql)

                   ## INSERINDO DADOS ##
# Primeira maneira de inserir dados
sql = """insert into TABELA(nome, idade, sexo, salario)
values('gustavo', '22', 'M', '4000')"""
try:
    # Executando o comando SQL
    cursor.execute(sql)
    # Fazendo mudanças na database
    db.commit()
except:
    # Desfazendo mudanças caso dê erro
    db.rollback()
# Desconectando
db.close()
```

```
# Segunda maneira de inserir dados (insert dinamico)
sql = "insert into TABELA(nome, idade, sexo, salario)
values ('%s', '%d', '%c', '%d')" % ('gustavo', 22, 'M',
4000)
try:
    cursor.execute(sql)
    db.commit()
except:
    db.rollback()
db.close()

# Terceira maneira de inserir dados (parametro direto)
nome = 'gustavo'
idade = 22
sexo = 'M'
salario = 4000
try:
    cursor.execute('insert into TABELA
values("%s", "%d", "%c", "%d")' %(nome, idade, sexo,
salario))
    db.commit()
except:
    db.rollback()
db.close()

                ## LEITURA DE DADOS ##
# Busca de empregados com salarios maior que 900
sql = "select * from TABELA where salario
> '%d'" %(900)
try:
    cursor.execute(sql)
    results = cursor.fetchall() # Buscando todas linhas
    for row in results:
        name = row[0]
        age = row[1]
        sex = row[2]
        income = row[3]
```

```
        print "Nome=%s, Idade=%d, Sexo=%s,
Salario=%d" %(name, age, sex, income)
except:
    print "Erro na busca."
db.close()

# Deletando os empregados com salarios menores que 900
sql = "delete from TABELA where salario < '%d'" %(900)

                ## MANIPULAÇÃO DE ERROS ##
# Os erros podem ser alertados como por exemplo:
...
except MySQLdb.Error, e:
    print "Erro executando \n " + sql
    print e
except MySQLdb.Warning, w:
    print w
```

```
# Tabela de erros
```

Exceção	Descrição
Warning	Usado para emitir mensagens não fatais. Subclasse de StandardError.
Error	Classe Base para erros. Subclasse de StandardError.
InterfaceError	Usado para erros no módulo de banco de dados, não do database. Suclasse de Error.
DatabaseError	Usado em erro da base de dados. Subclasse de Error.
DataError	Subclass of DatabaseError that refers to errors in the data.
OperationalError	Subclasse de DatabaseError que refere-se aos erros como perda de conexão. Esses erros são normalmente externos ao controle do script Python.
IntegrityError	Subclasse de DatabaseError para situações que danificariam a integridade relacional.
InternalError	Subclasse de DatabaseError que refere-se aos erros internos do módulo de database, como cursor não ativo.
ProgrammingError	Subclasse de DatabaseError que refere-se a erros como nome errado da tabela ou coisas que podem seguramente ser reprovados.
NotSupportedError	Subclasse de DatabaseError que refere-se a tentativa de chamadas não suportadas funcionalmente.