CS 327E Lecture 7

Shirley Cohen

October 5, 2016

Announcements

- Reminder: Lab 2 work next week
- Lab 2 specs & grading rubric: Monday
- Lab 2 setup instructions: http://tinyurl.com/hymam9a
- Format of final exam

Homework for Today

- Chapter 3 from the Data Wrangling book

Question 1

Which of the following data formats is **not** covered in the assigned chapter for today?

A. CSV

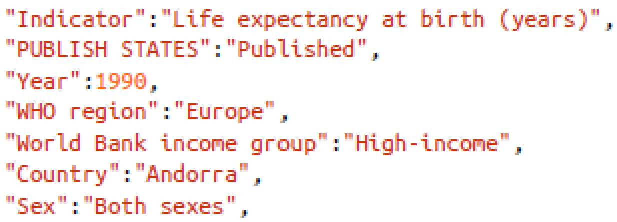
B. TSV

C. JSON

D. YAML

E. XML

Question 2



```
"Indicator":"Life expectancy at birth (years)",
"PUBLISH STATES":"Published",
"Year":1990,
"WHO region":"Europe",
"World Bank income group":"High-income",
"Country":"Andorra",
"Sex":"Both sexes",
```

The sample data shown above is in ___ format:

A. CSV

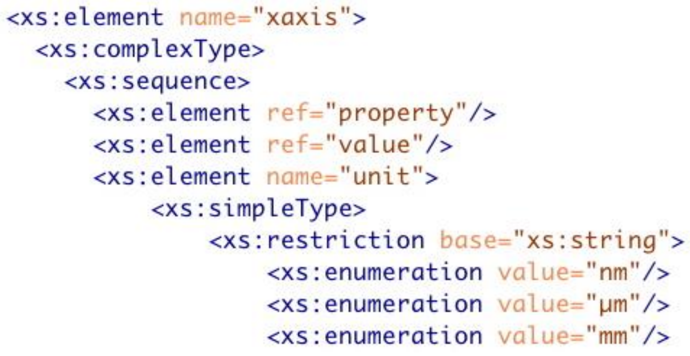
B. TSV

C. JSON

D. SQL

E. XML

Question 3



```
<xs:element name="xaxis">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="property"/>
      <xs:element ref="value"/>
      <xs:element name="unit">
          <xs:simpleType>
              <xs:restriction base="xs:string">
                  <xs:enumeration value="nm"/>
                  <xs:enumeration value="μm"/>
                  <xs:enumeration value="mm"/>
```

The sample data shown above is in ___ format:

A. CSV

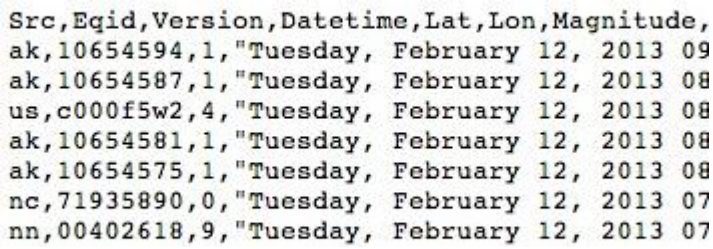
B. TSV

C. JSON

D. SQL

E. XML

Question 4



```
Src,Eqid,Version,Datetime,Lat,Lon,Magnitude,
ak,10654594,1,"Tuesday, February 12, 2013 09
ak,10654587,1,"Tuesday, February 12, 2013 08
us,c000f5w2,4,"Tuesday, February 12, 2013 08
ak,10654581,1,"Tuesday, February 12, 2013 08
ak,10654575,1,"Tuesday, February 12, 2013 08
nc,71935890,0,"Tuesday, February 12, 2013 07
nn,00402618,9,"Tuesday, February 12, 2013 07
```

The sample data shown above is in ___ format.

A. CSV

B. TSV

C. JSON

D. SQL

E. XML

Integrating with Python

- Python support for MySQL not build it. To interact with MySQL from Python, use a library called a “connector”

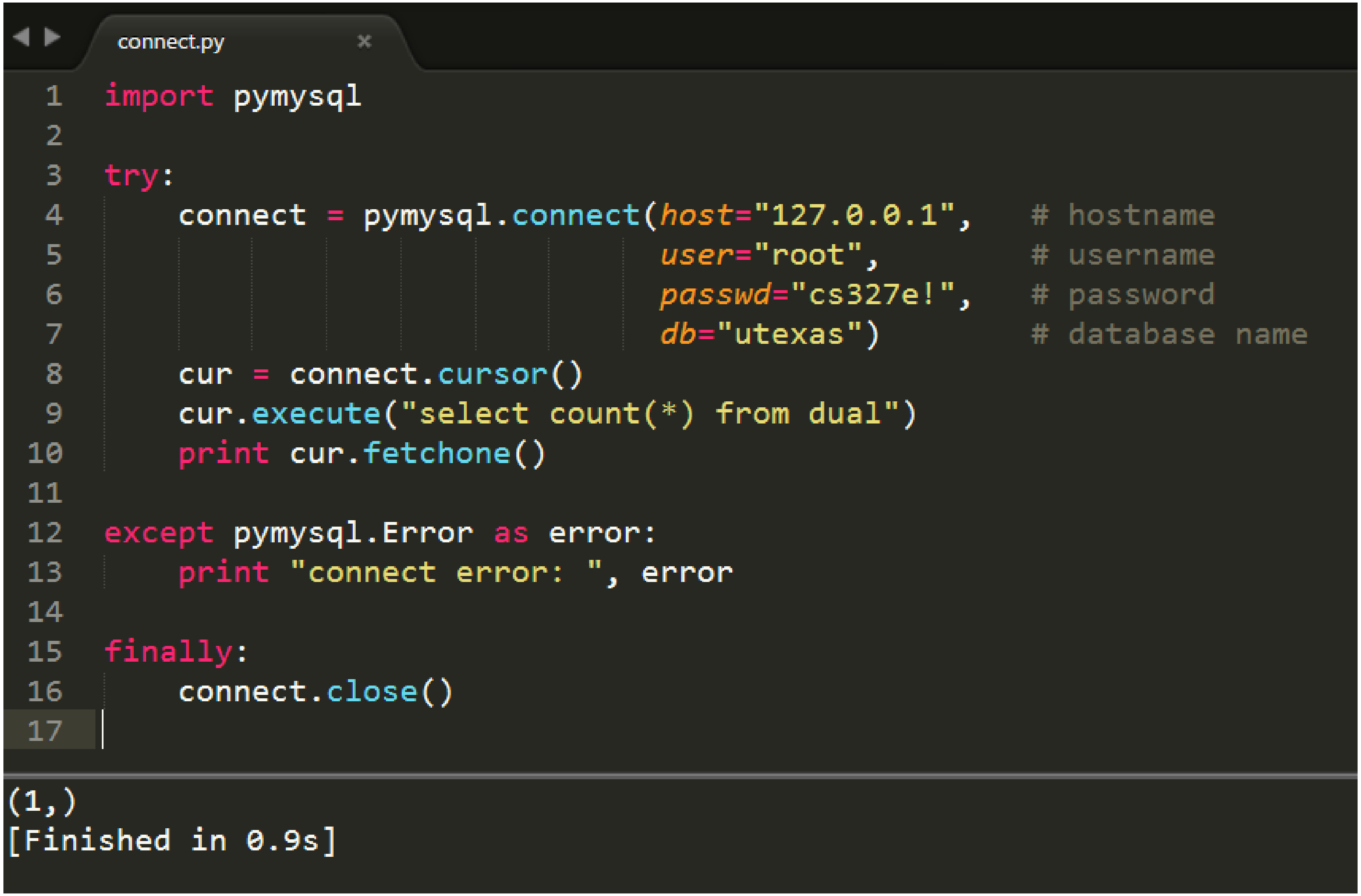
- PyMySQL connector: http://pymysql.readthedocs.io/en/latest/index.html

- Install PyMySQL through pip: `pip2 install pymysql`

- Assumes existing Python 2.7 installation:

```
 python –V
pip2 -V
```

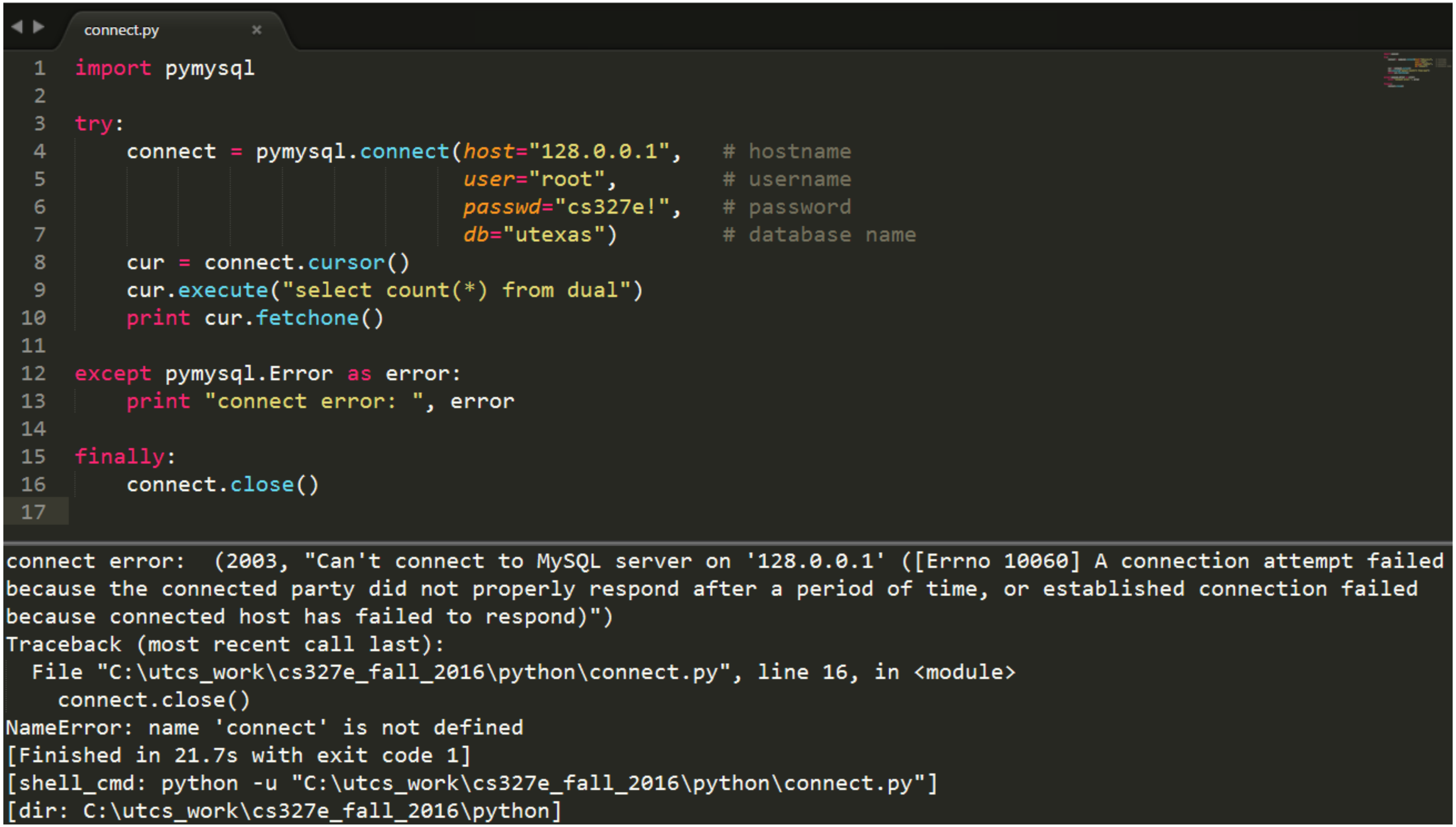
Connection Test



```
import pymysql

try:
    connect = pymysql.connect(host="127.0.0.1",   # hostname
                              user="root",        # username
                              passwd="cs327e!",   # password
                              db="utexas")        # database name
    cur = connect.cursor()
    cur.execute("select count(*) from dual")
    print cur.fetchone()

except pymysql.Error as error:
    print "connect error: ", error

finally:
    connect.close()
```

```
(1,)
[Finished in 0.9s]
```

What can go wrong



```
 1  import pymysql
 2
 3  try:
 4      connect = pymysql.connect(host="128.0.0.1",    # hostname
 5                                user="root",         # username
 6                                passwd="cs327e!",    # password
 7                                db="utexas")         # database name
 8      cur = connect.cursor()
 9      cur.execute("select count(*) from dual")
10      print cur.fetchone()
11
12  except pymysql.Error as error:
13      print "connect error: ", error
14
15  finally:
16      connect.close()
17
```

```
connect error:  (2003, "Can't connect to MySQL server on '128.0.0.1' ([Errno 10060] A connection attempt failed
because the connected party did not properly respond after a period of time, or established connection failed
because connected host has failed to respond)")
Traceback (most recent call last):
  File "C:\utcs_work\cs327e_fall_2016\python\connect.py", line 16, in <module>
    connect.close()
NameError: name 'connect' is not defined
[Finished in 21.7s with exit code 1]
[shell_cmd: python -u "C:\utcs_work\cs327e_fall_2016\python\connect.py"]
[dir: C:\utcs_work\cs327e_fall_2016\python]
```

Concept Question 1

What caused this connection error?

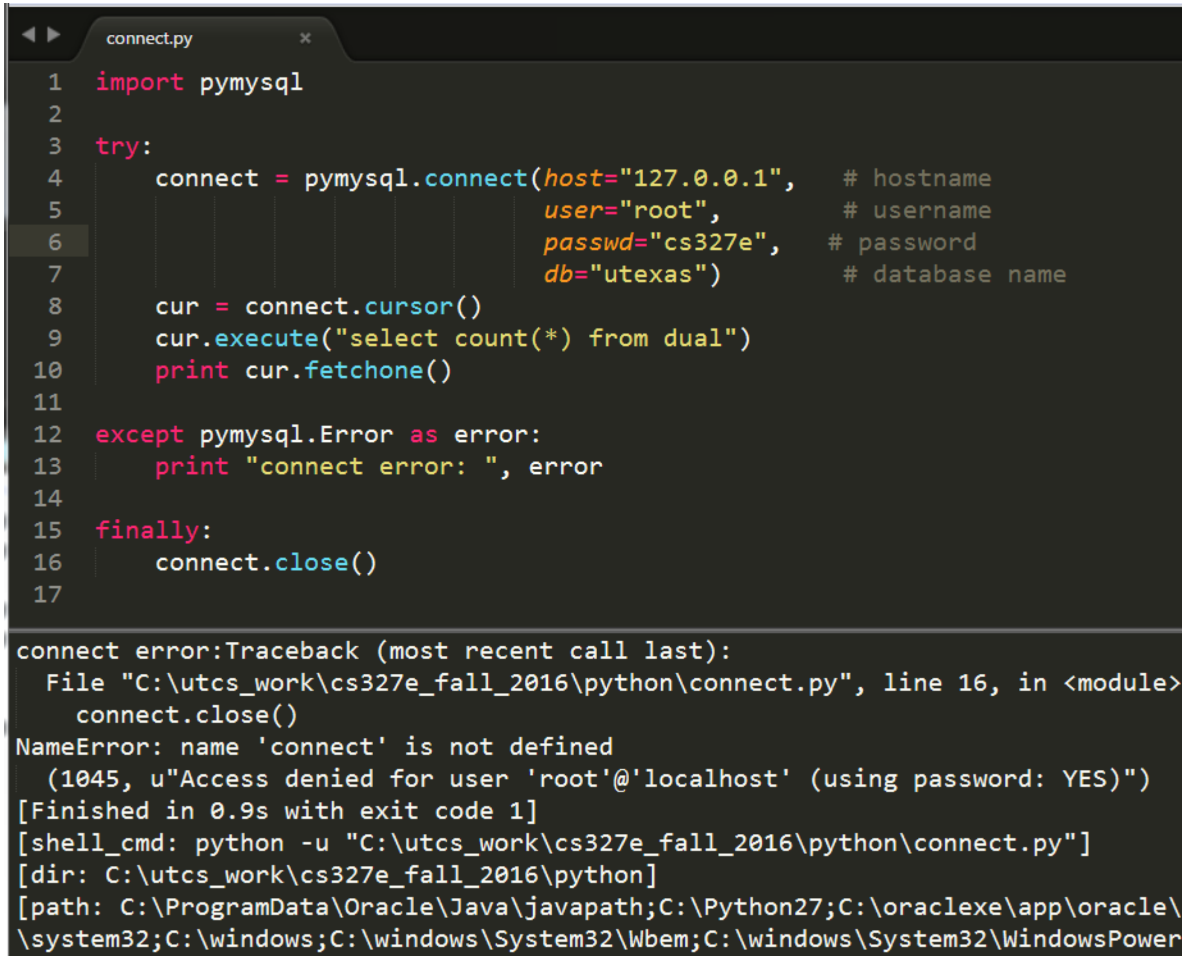
A. Bad host or IP address

B. Bad username and/or password

C. Bad db name

D. Bad SQL query

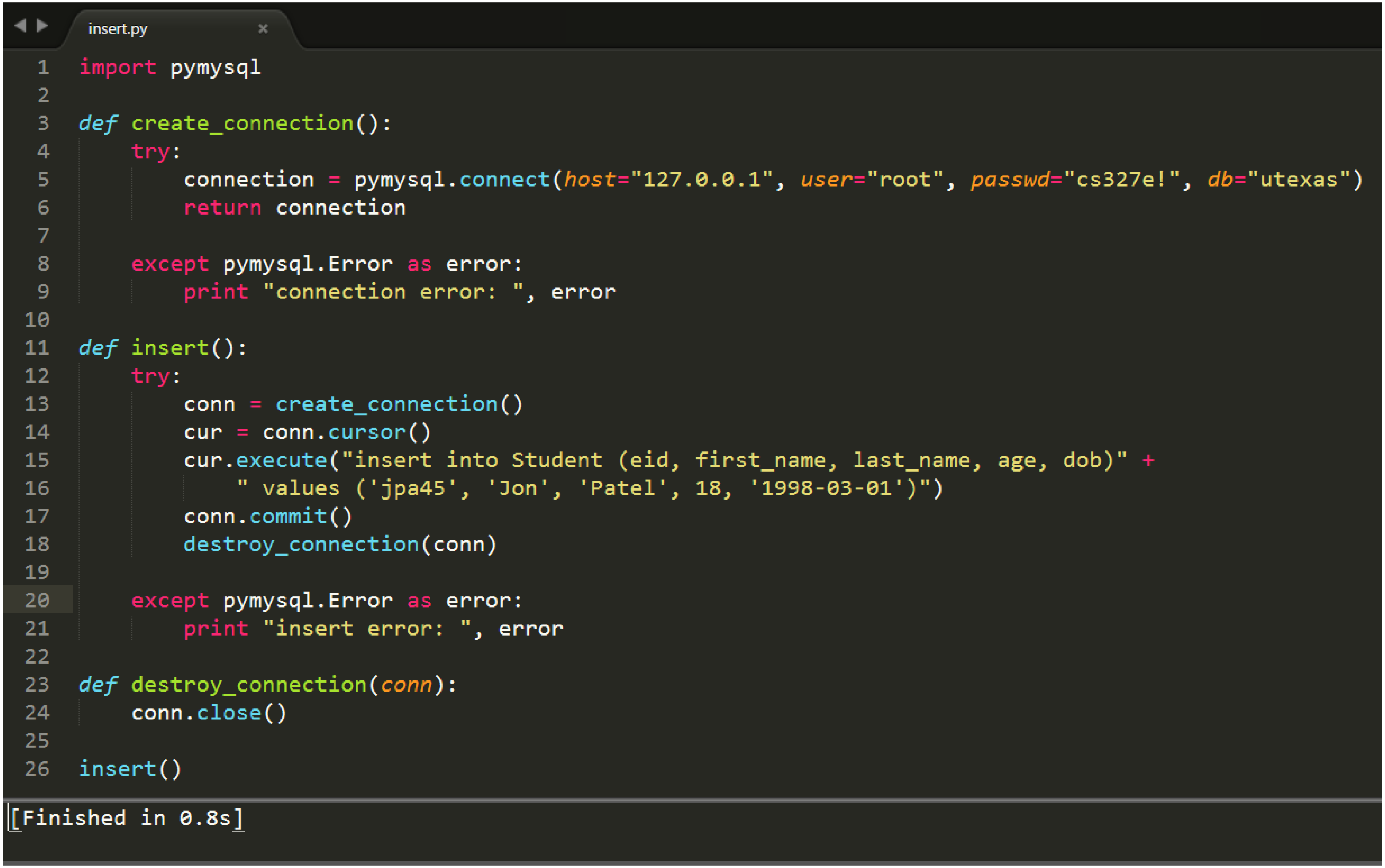
E. Any of the above



```
import pymysql

try:
    connect = pymysql.connect(host="127.0.0.1",   # hostname
                              user="root",        # username
                              passwd="cs327e",   # password
                              db="utexas")        # database name
    cur = connect.cursor()
    cur.execute("select count(*) from dual")
    print cur.fetchone()

except pymysql.Error as error:
    print "connect error: ", error

finally:
    connect.close()
```

```
connect error:Traceback (most recent call last):
  File "C:\utcs_work\cs327e_fall_2016\python\connect.py", line 16, in <module>
    connect.close()
NameError: name 'connect' is not defined
  (1045, u"Access denied for user 'root'@'localhost' (using password: YES)")
[Finished in 0.9s with exit code 1]
[shell_cmd: python -u "C:\utcs_work\cs327e_fall_2016\python\connect.py"]
[dir: C:\utcs_work\cs327e_fall_2016\python]
[path: C:\ProgramData\Oracle\Java\javapath;C:\Python27;C:\oraclexe\app\oracle\
\system32;C:\windows;C:\windows\System32\Wbem;C:\windows\System32\WindowsPower
```

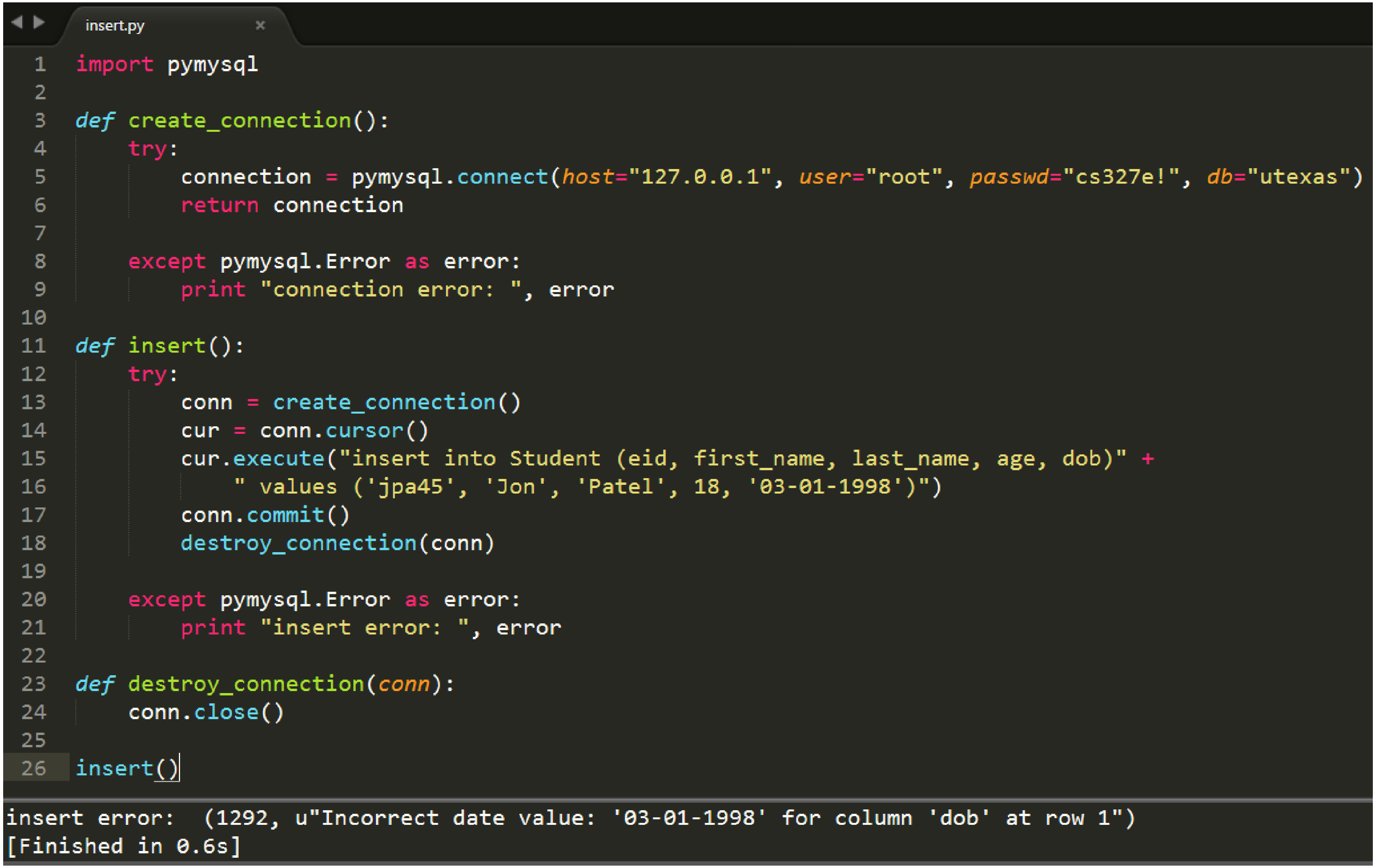
Single Insert



```python
import pymysql

def create_connection():
    try:
        connection = pymysql.connect(host="127.0.0.1", user="root", passwd="cs327e!", db="utexas")
        return connection

    except pymysql.Error as error:
        print "connection error: ", error

def insert():
    try:
        conn = create_connection()
        cur = conn.cursor()
        cur.execute("insert into Student (eid, first_name, last_name, age, dob)" +
            " values ('jpa45', 'Jon', 'Patel', 18, '1998-03-01')")
        conn.commit()
        destroy_connection(conn)

    except pymysql.Error as error:
        print "insert error: ", error

def destroy_connection(conn):
    conn.close()

insert()
```

```
[Finished in 0.8s]
```

What can go wrong

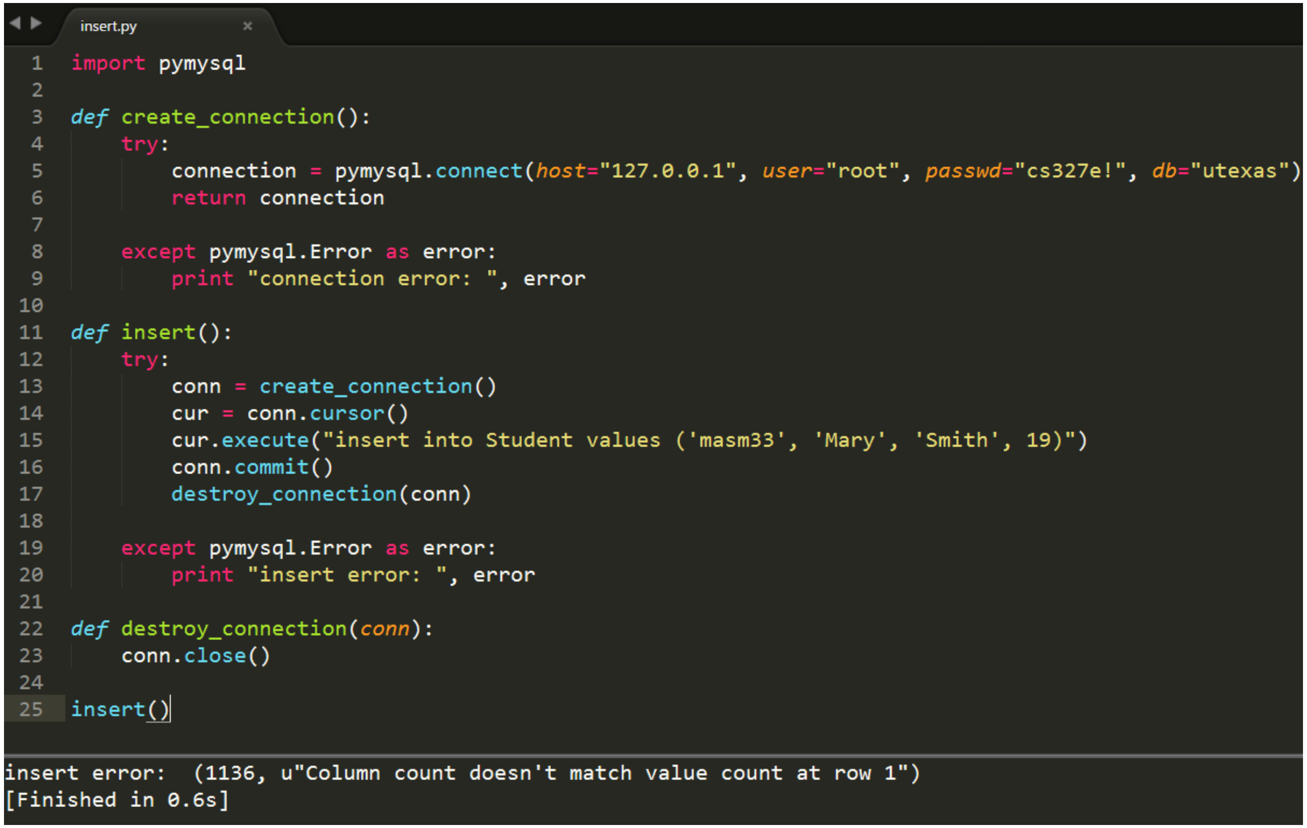


```
import pymysql

def create_connection():
    try:
        connection = pymysql.connect(host="127.0.0.1", user="root", passwd="cs327e!", db="utexas")
        return connection

    except pymysql.Error as error:
        print "connection error: ", error

def insert():
    try:
        conn = create_connection()
        cur = conn.cursor()
        cur.execute("insert into Student (eid, first_name, last_name, age, dob)" +
            " values ('jpa45', 'Jon', 'Patel', 18, '03-01-1998')")
        conn.commit()
        destroy_connection(conn)

    except pymysql.Error as error:
        print "insert error: ", error

def destroy_connection(conn):
    conn.close()

insert()
```

```
insert error:  (1292, u"Incorrect date value: '03-01-1998' for column 'dob' at row 1")
[Finished in 0.6s]
```

Concept Question 2

What caused this insert to fail?



```python
import pymysql

def create_connection():
    try:
        connection = pymysql.connect(host="127.0.0.1", user="root", passwd="cs327e!", db="utexas")
        return connection

    except pymysql.Error as error:
        print "connection error: ", error

def insert():
    try:
        conn = create_connection()
        cur = conn.cursor()
        cur.execute("insert into Student values ('masm33', 'Mary', 'Smith', 19)")
        conn.commit()
        destroy_connection(conn)

    except pymysql.Error as error:
        print "insert error: ", error

def destroy_connection(conn):
    conn.close()

insert()
```

```
insert error:  (1136, u"Column count doesn't match value count at row 1")
[Finished in 0.6s]
```

A. Duplicate record

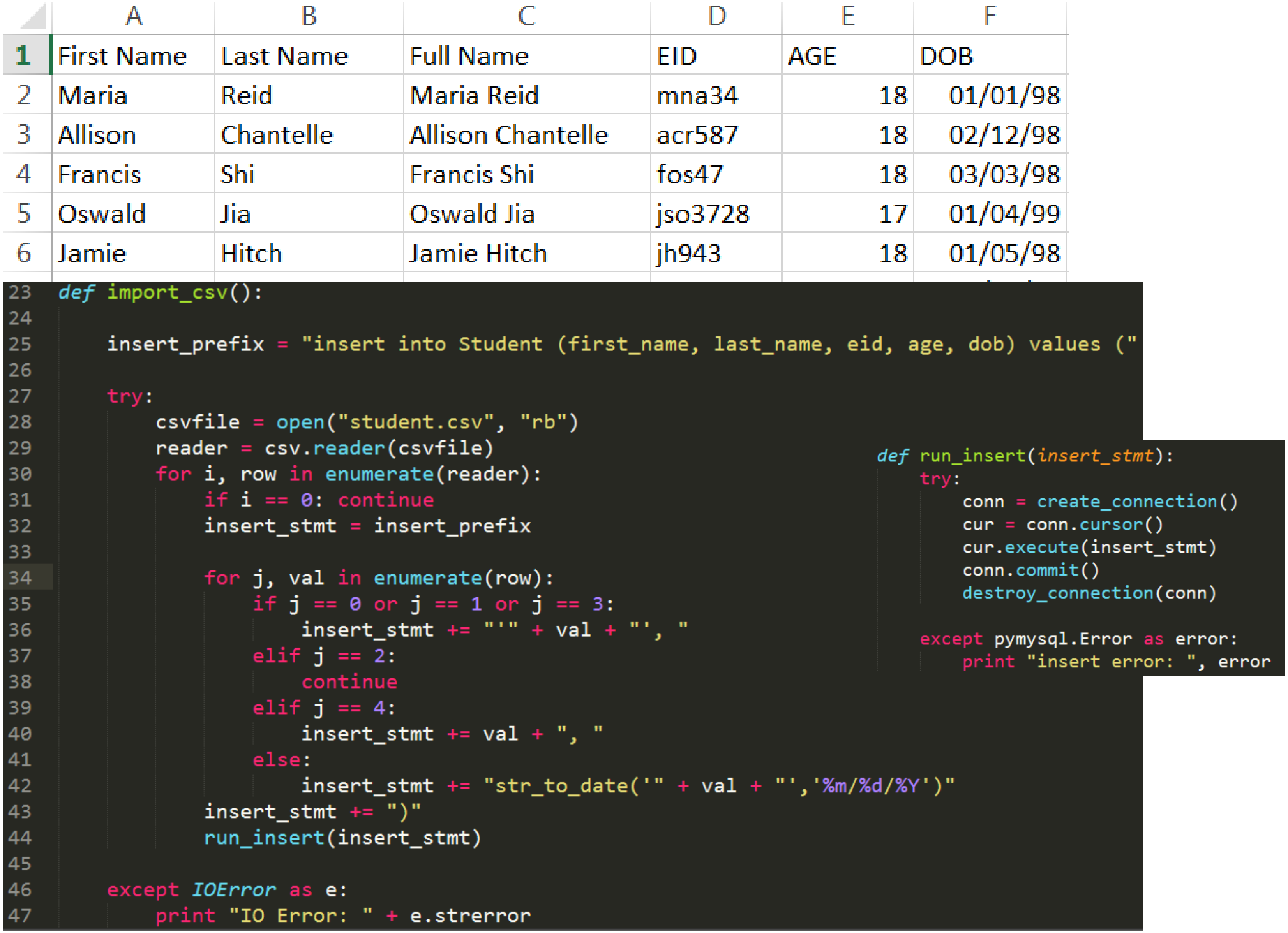
B. Insufficient values

C. Invalid connection or cursor object

D. Internal MySQL error

E. None of the above

Multiple Inserts

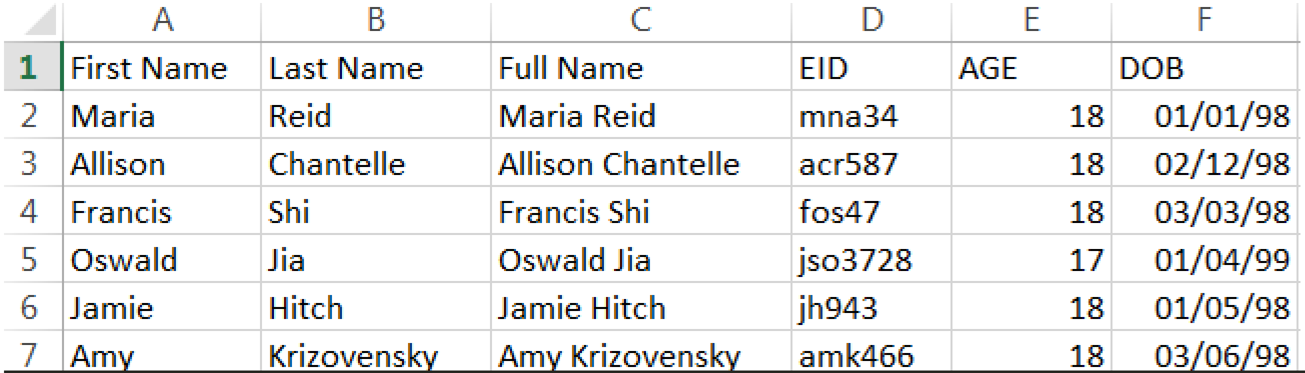


	A	B	C	D	E	F
1	First Name	Last Name	Full Name	EID	AGE	DOB
2	Maria	Reid	Maria Reid	mna34	18	01/01/98
3	Allison	Chantelle	Allison Chantelle	acr587	18	02/12/98
4	Francis	Shi	Francis Shi	fos47	18	03/03/98
5	Oswald	Jia	Oswald Jia	jso3728	17	01/04/99
6	Jamie	Hitch	Jamie Hitch	jh943	18	01/05/98

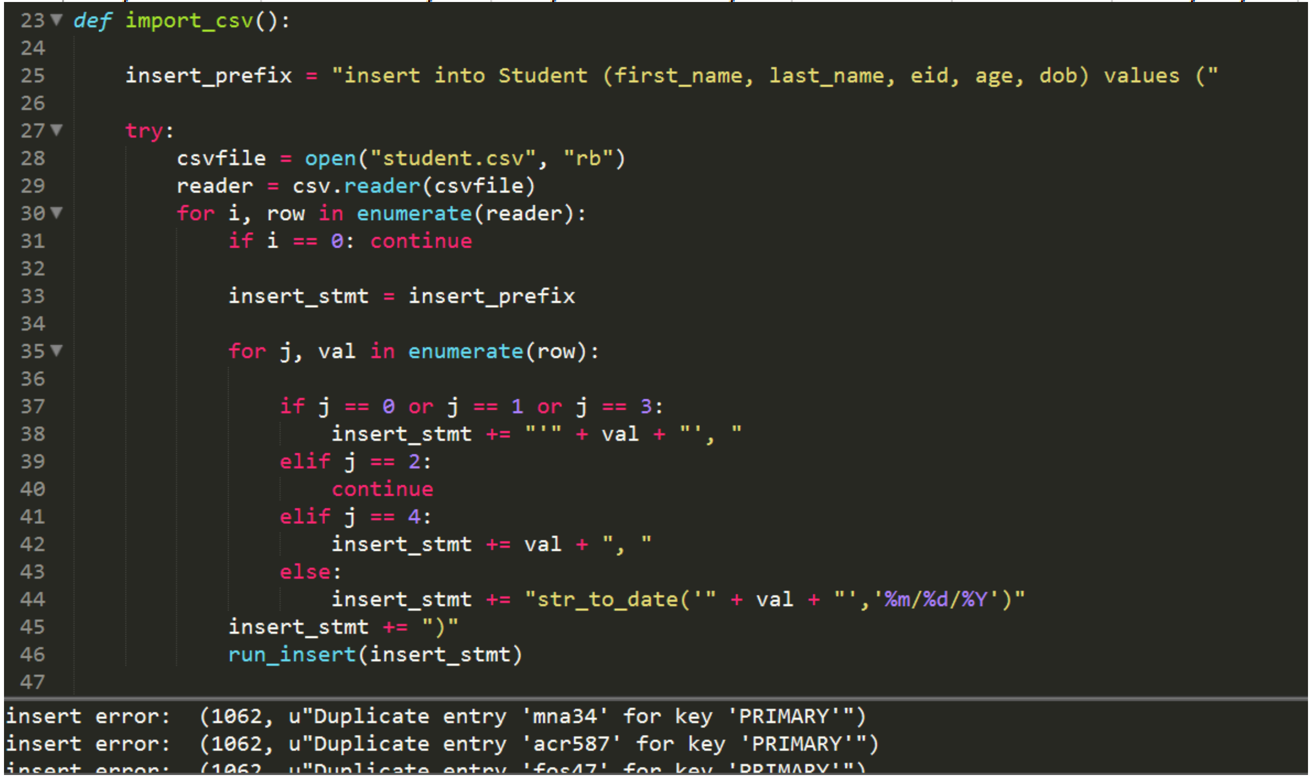
```python
def import_csv():

    insert_prefix = "insert into Student (first_name, last_name, eid, age, dob) values ("

    try:
        csvfile = open("student.csv", "rb")
        reader = csv.reader(csvfile)
        for i, row in enumerate(reader):
            if i == 0: continue
            insert_stmt = insert_prefix

            for j, val in enumerate(row):
                if j == 0 or j == 1 or j == 3:
                    insert_stmt += "'" + val + "', "
                elif j == 2:
                    continue
                elif j == 4:
                    insert_stmt += val + ", "
                else:
                    insert_stmt += "str_to_date('" + val + "','%m/%d/%Y')"
            insert_stmt += ")"
            run_insert(insert_stmt)

    except IOError as e:
        print "IO Error: " + e.strerror
```

```python
def run_insert(insert_stmt):
    try:
        conn = create_connection()
        cur = conn.cursor()
        cur.execute(insert_stmt)
        conn.commit()
        destroy_connection(conn)

    except pymysql.Error as error:
        print "insert error: ", error
```

What can go wrong



	A	B	C	D	E	F
1	First Name	Last Name	Full Name	EID	AGE	DOB
2	Maria	Reid	Maria Reid	mna34	18	01/01/98
3	Allison	Chantelle	Allison Chantelle	acr587	18	02/12/98
4	Francis	Shi	Francis Shi	fos47	18	03/03/98
5	Oswald	Jia	Oswald Jia	jso3728	17	01/04/99
6	Jamie	Hitch	Jamie Hitch	jh943	18	01/05/98
7	Amy	Krizovensky	Amy Krizovensky	amk466	18	03/06/98



```
def import_csv():

    insert_prefix = "insert into Student (first_name, last_name, eid, age, dob) values ("

    try:
        csvfile = open("student.csv", "rb")
        reader = csv.reader(csvfile)
        for i, row in enumerate(reader):
            if i == 0: continue

            insert_stmt = insert_prefix

            for j, val in enumerate(row):

                if j == 0 or j == 1 or j == 3:
                    insert_stmt += "'" + val + "', "
                elif j == 2:
                    continue
                elif j == 4:
                    insert_stmt += val + ", "
                else:
                    insert_stmt += "str_to_date('" + val + "','%m/%d/%Y')"
            insert_stmt += ")"
            run_insert(insert_stmt)

```

```
insert error:  (1062, u"Duplicate entry 'mna34' for key 'PRIMARY'")
insert error:  (1062, u"Duplicate entry 'acr587' for key 'PRIMARY'")
insert error:  (1062, u"Duplicate entry 'fos47' for key 'PRIMARY'")
```

Delete Statements

Option 1:

DELETE FROM T

e.g. DELETE FROM Student

Option 2:

DELETE FROM T WHERE c_0 = v_0

e.g. DELETE FROM Student WHERE eid = 'mna34'

Option 3:

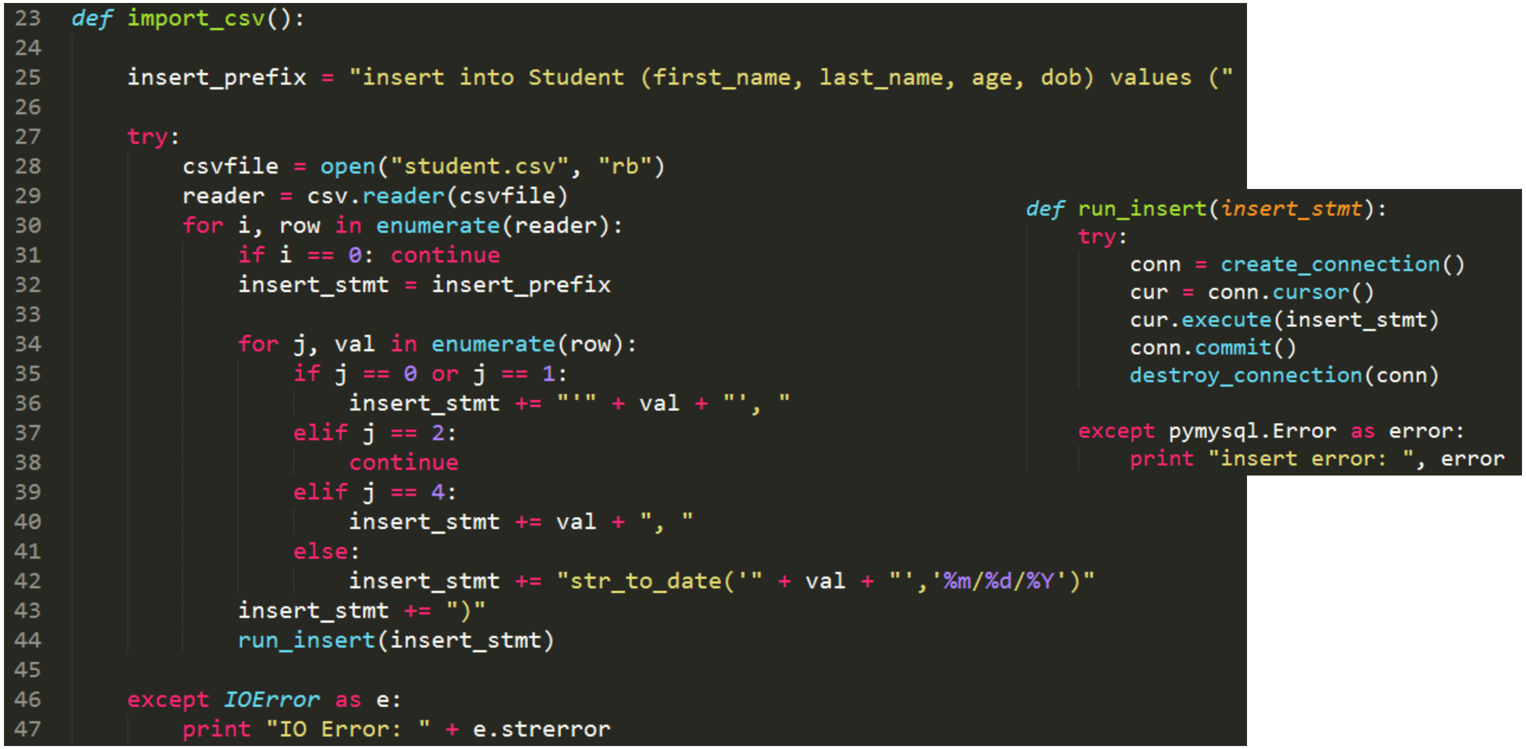
DELETE FROM T WHERE (SELECT * FROM T’)

e.g. DELETE FROM Current_Student WHERE (SELECT * FROM Archived_Student)

Note: T <> T’

Concept Question 3

Suppose we modify the PK in the Student table. Instead of the EID, we use an AUTO_INCREMENT column as the PK. What problem can arise from using a surrogate key?



```python
def import_csv():

    insert_prefix = "insert into Student (first_name, last_name, age, dob) values ("

    try:
        csvfile = open("student.csv", "rb")
        reader = csv.reader(csvfile)
        for i, row in enumerate(reader):
            if i == 0: continue
            insert_stmt = insert_prefix

            for j, val in enumerate(row):
                if j == 0 or j == 1:
                    insert_stmt += "'" + val + "', "
                elif j == 2:
                    continue
                elif j == 4:
                    insert_stmt += val + ", "
                else:
                    insert_stmt += "str_to_date('" + val + "','%m/%d/%Y')"
            insert_stmt += ")"
            run_insert(insert_stmt)

    except IOError as e:
        print "IO Error: " + e.strerror
```

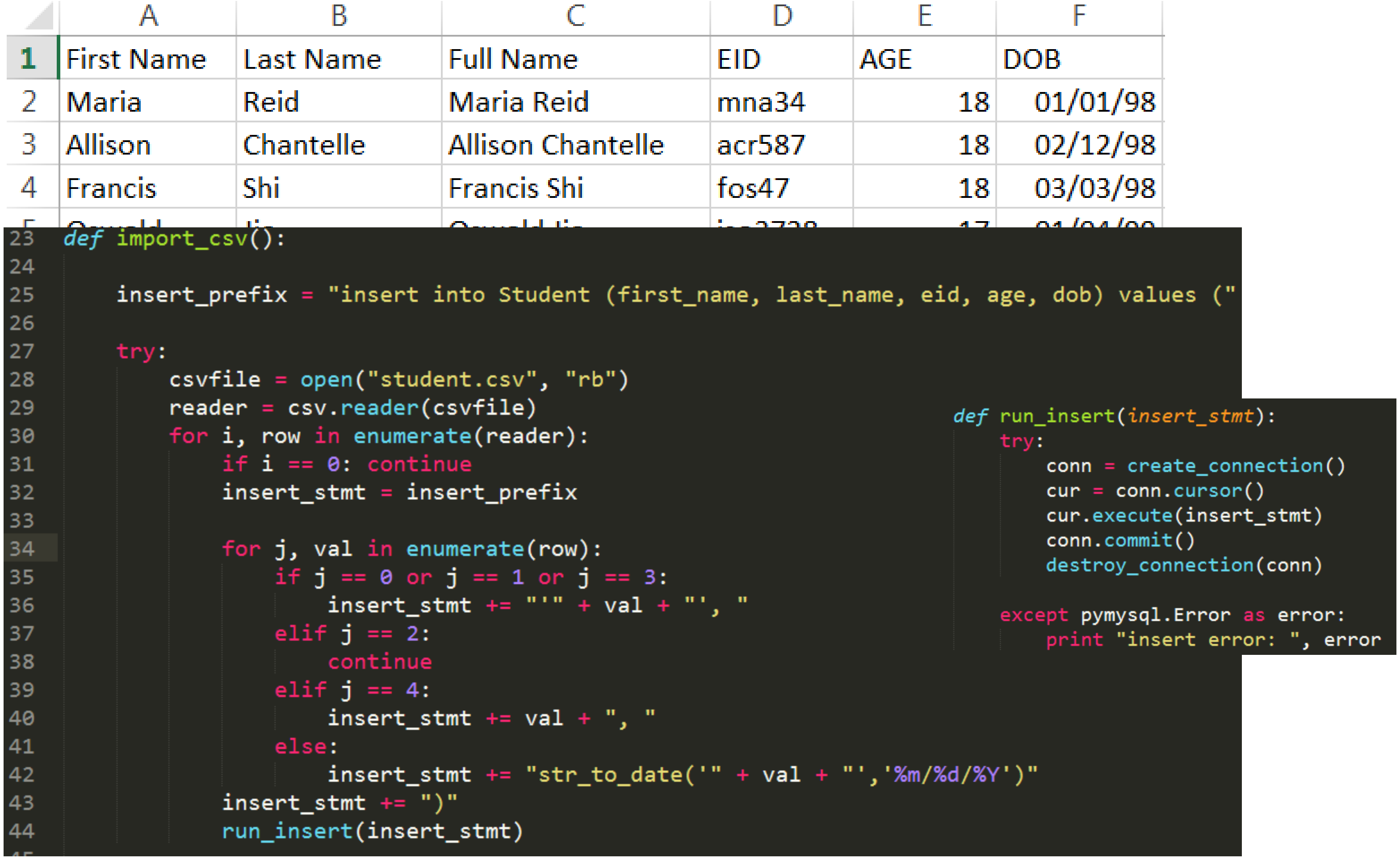
```python
def run_insert(insert_stmt):
    try:
        conn = create_connection()
        cur = conn.cursor()
        cur.execute(insert_stmt)
        conn.commit()
        destroy_connection(conn)

    except pymysql.Error as error:
        print "insert error: ", error
```

A. Surrogate keys are less descriptive
B. “Hidden” duplicate records

C. Can’t reset an AUTO_INCREMENT column
D. None of the above

Concept Question 4

Can we make this code run more efficiently? How so?



	A	B	C	D	E	F
1	First Name	Last Name	Full Name	EID	AGE	DOB
2	Maria	Reid	Maria Reid	mna34	18	01/01/98
3	Allison	Chantelle	Allison Chantelle	acr587	18	02/12/98
4	Francis	Shi	Francis Shi	fos47	18	03/03/98
5	Oswald	Jia	Oswald Jia	ioa2738	17	01/04/98

```
def import_csv():

    insert_prefix = "insert into Student (first_name, last_name, eid, age, dob) values ("

    try:
        csvfile = open("student.csv", "rb")
        reader = csv.reader(csvfile)
        for i, row in enumerate(reader):
            if i == 0: continue
            insert_stmt = insert_prefix

            for j, val in enumerate(row):
                if j == 0 or j == 1 or j == 3:
                    insert_stmt += "'" + val + "', "
                elif j == 2:
                    continue
                elif j == 4:
                    insert_stmt += val + ", "
                else:
                    insert_stmt += "str_to_date('" + val + "','%m/%d/%Y')"
            insert_stmt += ")"
            run_insert(insert_stmt)
```

```
def run_insert(insert_stmt):
    try:
        conn = create_connection()
        cur = conn.cursor()
        cur.execute(insert_stmt)
        conn.commit()
        destroy_connection(conn)

    except pymysql.Error as error:
        print "insert error: ", error
```

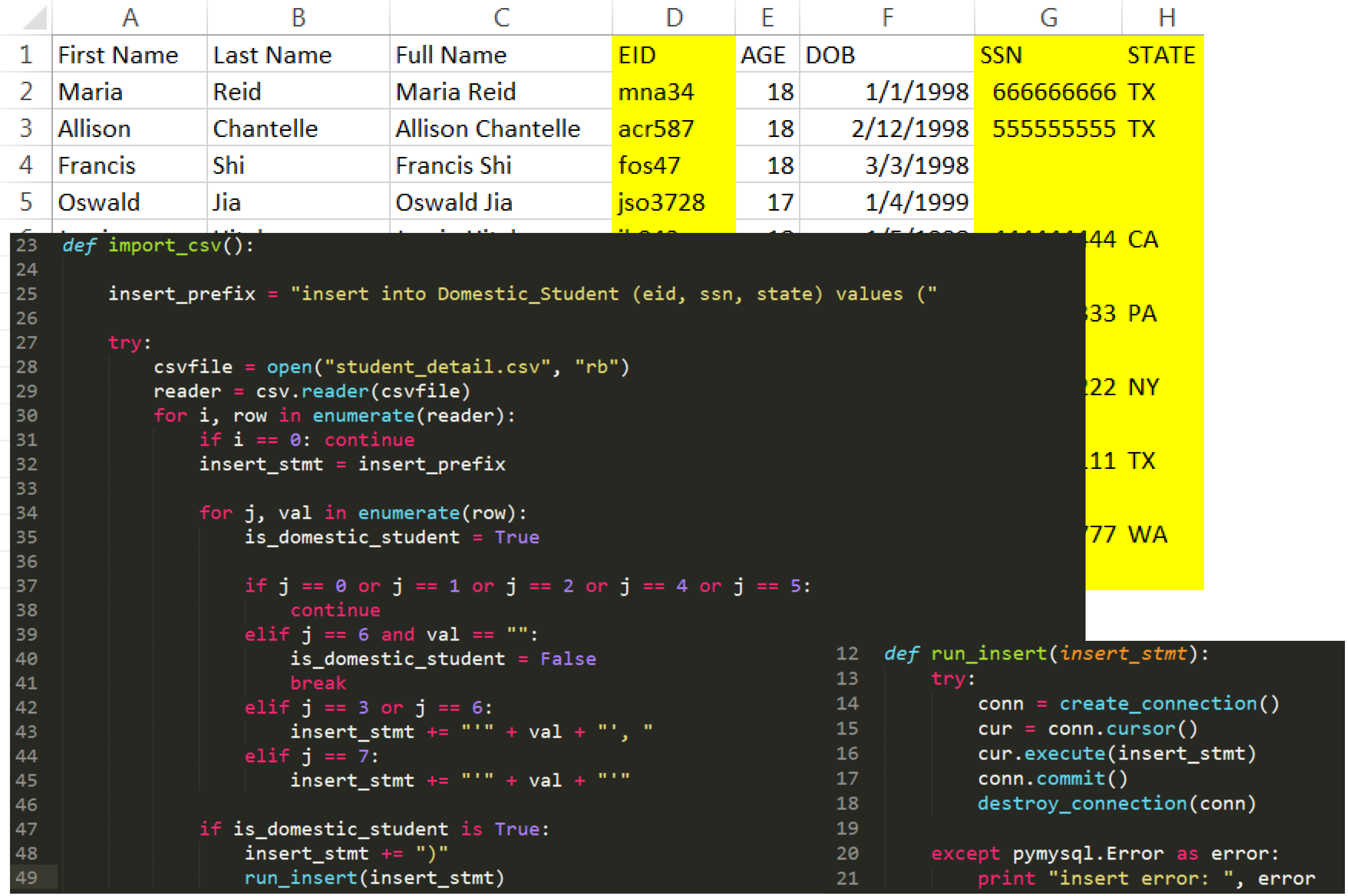
A. Reuse the connection

B. Commit inserts in batches

C. Remove print statements

D. All of the above

Inserts with FKs



	A	B	C	D	E	F	G	H
1	First Name	Last Name	Full Name	EID	AGE	DOB	SSN	STATE
2	Maria	Reid	Maria Reid	mna34	18	1/1/1998	666666666	TX
3	Allison	Chantelle	Allison Chantelle	acr587	18	2/12/1998	555555555	TX
4	Francis	Shi	Francis Shi	fos47	18	3/3/1998		
5	Oswald	Jia	Oswald Jia	jso3728	17	1/4/1999		

44 CA

33 PA

22 NY

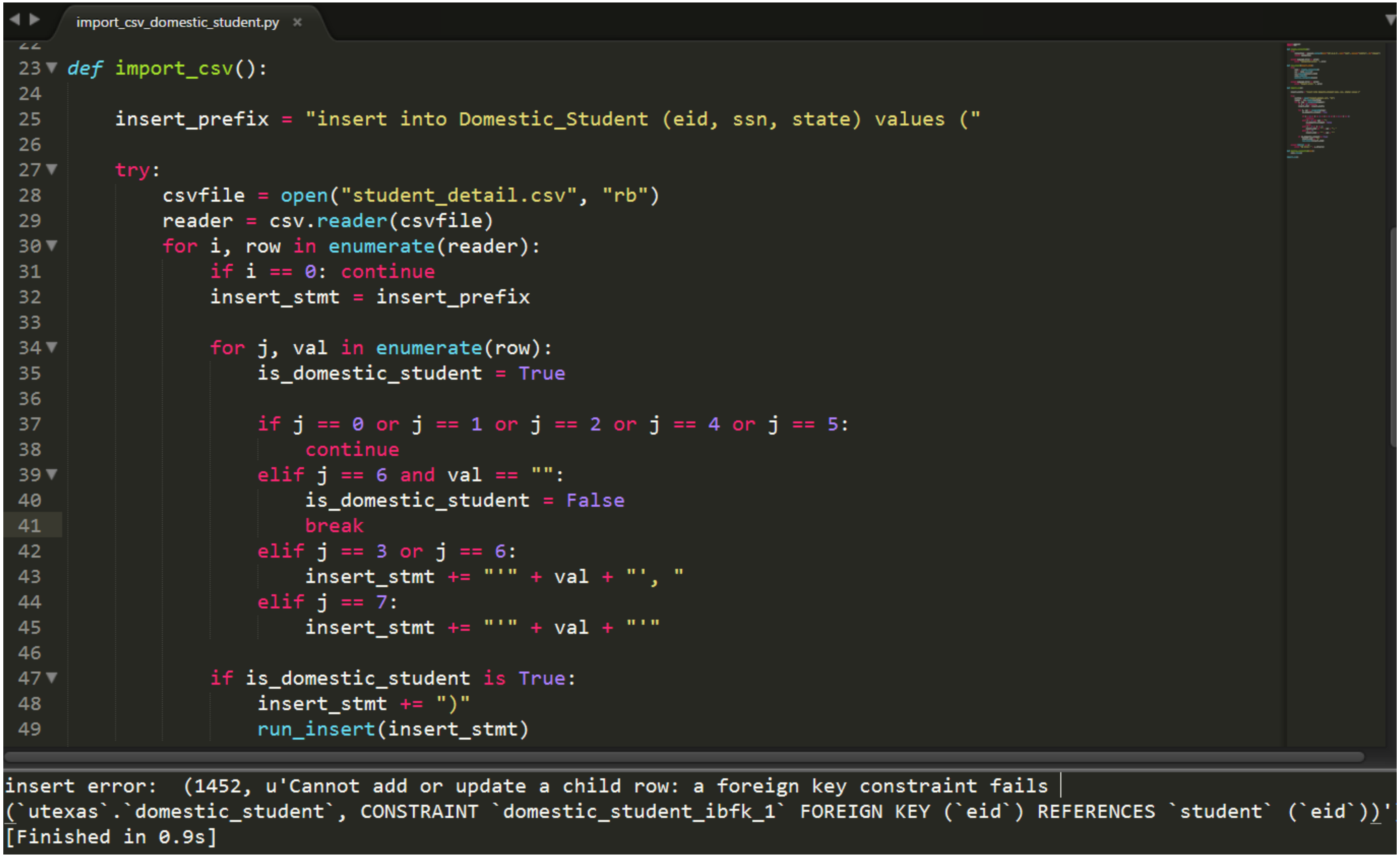
11 TX

77 WA

```python
def import_csv():

    insert_prefix = "insert into Domestic_Student (eid, ssn, state) values ("

    try:
        csvfile = open("student_detail.csv", "rb")
        reader = csv.reader(csvfile)
        for i, row in enumerate(reader):
            if i == 0: continue
            insert_stmt = insert_prefix

            for j, val in enumerate(row):
                is_domestic_student = True

                if j == 0 or j == 1 or j == 2 or j == 4 or j == 5:
                    continue
                elif j == 6 and val == "":
                    is_domestic_student = False
                    break
                elif j == 3 or j == 6:
                    insert_stmt += "'" + val + "', "
                elif j == 7:
                    insert_stmt += "'" + val + "'"

            if is_domestic_student is True:
                insert_stmt += ")"
                run_insert(insert_stmt)
```

```python
def run_insert(insert_stmt):
    try:
        conn = create_connection()
        cur = conn.cursor()
        cur.execute(insert_stmt)
        conn.commit()
        destroy_connection(conn)

    except pymysql.Error as error:
        print "insert error: ", error
```

What can go wrong



```python
def import_csv():

    insert_prefix = "insert into Domestic_Student (eid, ssn, state) values ("

    try:
        csvfile = open("student_detail.csv", "rb")
        reader = csv.reader(csvfile)
        for i, row in enumerate(reader):
            if i == 0: continue
            insert_stmt = insert_prefix

            for j, val in enumerate(row):
                is_domestic_student = True

                if j == 0 or j == 1 or j == 2 or j == 4 or j == 5:
                    continue
                elif j == 6 and val == "":
                    is_domestic_student = False
                    break
                elif j == 3 or j == 6:
                    insert_stmt += "'" + val + "', "
                elif j == 7:
                    insert_stmt += "'" + val + "'"

            if is_domestic_student is True:
                insert_stmt += ")"
                run_insert(insert_stmt)
```

```
insert error:  (1452, u'Cannot add or update a child row: a foreign key constraint fails
(`utexas`.`domestic_student`, CONSTRAINT `domestic_student_ibfk_1` FOREIGN KEY (`eid`) REFERENCES `student` (`eid`))'
[Finished in 0.9s]
```

Final Remarks

- Avoid using surrogate keys. If you have no choice, check for duplicate records by manually inspecting the data. We will learn a more efficient way to do this when we cover the GROUP-BY clause
- Read the API docs for PyMySQL: http://pymysql.readthedocs.io/en/latest/index.html
- PyMySQL sample code available in our snippets repo
- Please setup your environment prior to Monday’s class (and if you get stuck, post the issue on Piazza)