pandas 라이브러리와 pymysql

1. read_sql()
    - sql 연결객체를 활용하여 쿼리 구문으로 반환된 결과를 데이터프레임으로 바로 생성해 주는 함수

- 테이블 생성

[ ]:
```
USE student_mgmt;
DROP TABLE IF EXISTS students;
CREATE TABLE students (
  id TINYINT NOT NULL AUTO_INCREMENT,
  name VARCHAR(10) NOT NULL,
  gender ENUM('man','woman') NOT NULL,
  birth DATE NOT NULL,
  english TINYINT NOT NULL,
  math TINYINT NOT NULL,
  korean TINYINT NOT NULL,
  PRIMARY KEY (id)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

[ ]:
```

```

- 데이터 입력

[ ]:
```
INSERT INTO students (name, gender, birth, english, math, korean) VALUES ('dave', 'man', '1983-07
INSERT INTO students (name, gender, birth, english, math, korean) VALUES ('minsun', 'woman', '198
INSERT INTO students (name, gender, birth, english, math, korean) VALUES ('david', 'man', '1982-1
INSERT INTO students (name, gender, birth, english, math, korean) VALUES ('jade', 'man', '1979-11
INSERT INTO students (name, gender, birth, english, math, korean) VALUES ('jane', 'man', '1990-11
INSERT INTO students (name, gender, birth, english, math, korean) VALUES ('wage', 'woman', '1982-
INSERT INTO students (name, gender, birth, english, math, korean) VALUES ('tina', 'woman', '1982-
```

Type *Markdown* and LaTeX: α^2

- read_sql()
- sql 연결객체를 활용하여 쿼리 구문으로 반환된 결과를 데이터프레임으로 바로 생성해 주는 함수

[13]:
```python
import pymysql
import pandas as pd
```

[14]:
```python
host_name = 'localhost'
host_port = 3306
username = 'root'
password = 'toor'
database_name = 'student_mgmt'
```

[15]:
```python
# db 연결
db = pymysql.connect(
    host=host_name,     # MySQL Server Address
    port=host_port,          # MySQL Server Port
    user=username,      # MySQL username
    passwd=password,    # password for MySQL username
    db=database_name,   # Database name
    charset='utf8'
)
```

pandas.read_sql(쿼리, 연결된 db connection 객체)

[16]:
```python
sql = "show tables"
```

[17]:
```python
df = pd.read_sql(sql, db)
```

[18]:
```python
df
```

	Tables_in_student_mgmt
0	students

[19]:
```python
sql = "select * from students"
df = pd.read_sql(sql,db)
```

```
C:₩ProgramData₩Anaconda3₩lib₩site-packages₩pandas₩io₩sql.py:761: UserWarning: pandas only support SQL
Alchemy connectable(engine/connection) ordatabase string URI or sqlite3 DBAPI2 connectionother DBAPI2
objects are not tested, please consider using SQLAlchemy
  warnings.warn(
```

[20]:
```python
df
```

	id	name	gender	birth	english	math	korean
0	1	dave	man	1983-07-16	90	80	71
1	2	minsun	woman	1982-10-16	30	88	60
2	3	david	man	1982-12-10	78	77	30
3	4	jade	man	1979-11-01	45	66	20
4	5	jane	man	1990-11-12	65	32	90
5	6	wage	woman	1982-01-13	76	30	80
6	7	tina	woman	1982-12-03	87	62	71

[21]:
```python
type(df['math'][0]) # 테이블의 컬럼 형식을 그대로 유지
```

```
numpy.int64
```

[22]:
```
df.to_csv('students.csv', sep=',', index=False, encoding='utf-8')
df
```

```
   id    name gender       birth  english  math  korean
0   1    dave    man  1983-07-16       90    80      71
1   2  minsun  woman  1982-10-16       30    88      60
2   3   david    man  1982-12-10       78    77      30
3   4    jade    man  1979-11-01       45    66      20
4   5    jane    man  1990-11-12       65    32      90
5   6    wage  woman  1982-01-13       76    30      80
6   7    tina  woman  1982-12-03       87    62      71
```

[23]:
```
db.close()
```

Type *Markdown* and LaTeX: α^2

Type *Markdown* and LaTeX: α^2

외래키(FOREIGN KEY)를 만드는 이유

- 두 테이블 사이에 관계를 선언해서, 데이터의 무결성을 보장

[24]:
```
import pymysql
import pandas as pd
```

[25]:
```
host_name = 'localhost'
host_port = 3306
username = 'root'
password = 'toor'
database_name = 'sqlDB'
```

[26]:
```
db = pymysql.connect(
    host=host_name,     # MySQL Server Address
    port=host_port,          # MySQL Server Port
    user=username,      # MySQL username
    passwd=password,    # password for MySQL username
    db=database_name,   # Database name
    charset='utf8'
)
```

[27]:
```
sql = "select * from userTbl"
df = pd.read_sql(sql,db)
df
```

```
C:₩ProgramData₩Anaconda3₩lib₩site-packages₩pandas₩io₩sql.py:761: UserWarning: pandas only support SQL
Alchemy connectable(engine/connection) ordatabase string URI or sqlite3 DBAPI2 connectionother DBAPI2
objects are not tested, please consider using SQLAlchemy
  warnings.warn(
```

	userID	name	birthYear	addr	mobile1	mobile2	height	mDate
0	BBK	바비킴	1973	서울	010	0000000	176	2013-05-05
1	EJW	은지원	1972	경북	011	8888888	174	2014-03-03
2	JKW	조관우	1965	경기	018	9999999	172	2010-10-10
3	JYP	조용필	1950	경기	011	4444444	166	2009-04-04
4	KBS	김범수	1979	경남	011	2222222	173	2012-04-04
5	KKH	김경호	1971	전남	019	3333333	177	2007-07-07
6	LJB	임재범	1963	서울	016	6666666	182	2009-09-09
7	LSG	이승기	1987	서울	011	1111111	182	2008-08-08
8	SSK	성시경	1979	서울	None	None	186	2013-12-12
9	YJS	윤종신	1969	경남	None	None	170	2005-05-05

[28]:
```python
sql = "select * from buyTbl"
df = pd.read_sql(sql,db)
df
```

```
C:₩ProgramData₩Anaconda3₩lib₩site-packages₩pandas₩io₩sql.py:761: UserWarning: pandas only support SQL
Alchemy connectable(engine/connection) ordatabase string URI or sqlite3 DBAPI2 connectionother DBAPI2
objects are not tested, please consider using SQLAlchemy
  warnings.warn(
```

	num	userID	prodName	groupName	price	amount
0	1	KBS	운동화	None	30	2
1	2	KBS	노트북	전자	1000	1
2	3	JYP	모니터	전자	200	1
3	4	BBK	모니터	전자	200	5
4	5	KBS	청바지	의류	50	3
5	6	BBK	메모리	전자	80	10
6	7	SSK	책	서적	15	5
7	8	EJW	책	서적	15	2
8	9	EJW	청바지	의류	50	1
9	10	BBK	운동화	None	30	2
10	11	EJW	책	서적	15	1
11	12	BBK	운동화	None	30	2

buyTbl에 데이터를 추가

- 외래키로 지정되어 있는 userID에 입력되는 새로운 값 STJ가 userTbl에 없는 값이어서 무결성 오류 발생

[29]:

```
cursor = db.cursor()
sql_query = "INSERT INTO buyTbl (userID, prodName, groupName, price, amount) VALUES('STJ', '운동
cursor.execute(sql_query)
db.commit()
```

```
---------------------------------------------------------------------------
IntegrityError                            Traceback (most recent call last)
Input In [29], in <cell line: 3>()
      1 cursor = db.cursor()
      2 sql_query = "INSERT INTO buyTbl (userID, prodName, groupName, price, amount) VALUES('STJ',
'운동화', '의류', 30, 2);"
----> 3 cursor.execute(sql_query)
      4 db.commit()

File C:\ProgramData\Anaconda3\lib\site-packages\pymysql\cursors.py:148, in Cursor.execute(self,
query, args)
    144     pass
    146 query = self.mogrify(query, args)
--> 148 result = self._query(query)
    149 self._executed = query
    150 return result

File C:\ProgramData\Anaconda3\lib\site-packages\pymysql\cursors.py:310, in Cursor._query(self,
q)
    308 self._last_executed = q
    309 self._clear_result()
--> 310 conn.query(q)
    311 self._do_get_result()
    312 return self.rowcount

File C:\ProgramData\Anaconda3\lib\site-packages\pymysql\connections.py:548, in Connection.query
(self, sql, unbuffered)
    546     sql = sql.encode(self.encoding, "surrogateescape")
    547 self._execute_command(COMMAND.COM_QUERY, sql)
--> 548 self._affected_rows = self._read_query_result(unbuffered=unbuffered)
    549 return self._affected_rows

File C:\ProgramData\Anaconda3\lib\site-packages\pymysql\connections.py:775, in Connection._read
_query_result(self, unbuffered)
    773 else:
    774     result = MySQLResult(self)
```

```
--> 775     result.read()
    776 self._result = result
    777 if result.server_status is not None:

File C:₩ProgramData₩Anaconda3₩lib₩site-packages₩pymysql₩connections.py:1156, in MySQLResult.rea
d(self)
   1154 def read(self):
   1155     try:
-> 1156         first_packet = self.connection._read_packet()
   1158         if first_packet.is_ok_packet():
   1159             self._read_ok_packet(first_packet)

File C:₩ProgramData₩Anaconda3₩lib₩site-packages₩pymysql₩connections.py:725, in Connection._read
_packet(self, packet_type)
    723     if self._result is not None and self._result.unbuffered_active is True:
    724         self._result.unbuffered_active = False
--> 725     packet.raise_for_error()
    726 return packet

File C:₩ProgramData₩Anaconda3₩lib₩site-packages₩pymysql₩protocol.py:221, in MysqlPacket.raise_f
or_error(self)
    219 if DEBUG:
    220     print("errno =", errno)
--> 221 err.raise_mysql_exception(self._data)

File C:₩ProgramData₩Anaconda3₩lib₩site-packages₩pymysql₩err.py:143, in raise_mysql_exception(da
ta)
    141 if errorclass is None:
    142     errorclass = InternalError if errno < 1000 else OperationalError
--> 143 raise errorclass(errno, errval)

IntegrityError: (1452, 'Cannot add or update a child row: a foreign key constraint fails (`sqldb`.`b
uytbl`, CONSTRAINT `buytbl_ibfk_1` FOREIGN KEY (`userID`) REFERENCES `usertbl` (`userID`))')
```

에러가 나면 정상임

- CONSTRAINT buyTbl_ibfk_1 FOREIGN KEY (userID) REFERENCES userTbl (userID)
- userTbl 에 userID가 STJ인 데이터가 없기 때문에,
    - FOREIGN KEY (userID) REFERENCES userTbl(userID)
    - buyTbl 테이블의 userID 커럼은 userTbl 테이블의 userID를 참조할 때, userTbl 테이블에 userID가 STJ인 데이터가 없으면, 입력이 안됨
    - 데이터 무결성 (두 테이블간 관계에 있어서, 데이터의 정확성을 보장하는 제약 조건을 넣는 것임)
    - 현업에서는 꼭 필요한 경우만 사용하는 경우가 많음 (비즈니스 로직이 다양하기 때문에, 제약을 걸어놓을 경우, 예외적인 비즈니스 로직 처리가 어렵기 때문)

[30]:
```python
cursor = db.cursor()
SQL_QUERY = "INSERT INTO buyTbl (userID, prodName, groupName, price, amount) VALUES('BBK', '운동
cursor.execute(SQL_QUERY)
db.commit()
```

[31]:
```python
db.close()
```

[32]:
```python
# db 연결을 활성화 해주는 함수 구현
def conn(d_name) :
    import pymysql
    host_name = 'localhost'
    host_port = 3306
    username = 'root'
    password = 'toor'
    database_name = d_name
    db = pymysql.connect(
        host=host_name,     # MySQL Server Address
        port=host_port,          # MySQL Server Port
        user=username,      # MySQL username
        passwd=password,    # password for MySQL username
        db=database_name,   # Database name
        charset='utf8'
    )
    return db
```

[33]:
```
db = conn('sqlDB')
```

이번에는 userTbl 에 userID가 STJ 인 데이터를 넣어준 후에, 다시 buyTbl userID에 STJ 관련 데이터를 넣어줌

[34]:
```
cursor = db.cursor()
sql_query = "INSERT INTO userTbl VALUES('STJ', '서태지', 1975, '경기', '011', '00000000', 171, '2
cursor.execute(sql_query)
db.commit()
```

[35]:
```
SQL_QUERY = "INSERT INTO buyTbl (userID, prodName, groupName, price, amount) VALUES('STJ', '운동
cursor.execute(SQL_QUERY)
db.commit()
```

이번에는 userTbl에 userID가 STJ 관련 데이터를 삭제해봄

[36]:

```
sql_query = "delete from userTbl where userID='STJ'"
cursor.execute(sql_query)
db.commit()
```

```
---------------------------------------------------------------------------
IntegrityError                            Traceback (most recent call last)
Input In [36], in <cell line: 2>()
      1 sql_query = "delete from userTbl where userID='STJ'"
----> 2 cursor.execute(sql_query)
      3 db.commit()

File C:₩ProgramData₩Anaconda3₩lib₩site-packages₩pymysql₩cursors.py:148, in Cursor.execute(self,
query, args)
    144     pass
    146 query = self.mogrify(query, args)
--> 148 result = self._query(query)
    149 self._executed = query
    150 return result

File C:₩ProgramData₩Anaconda3₩lib₩site-packages₩pymysql₩cursors.py:310, in Cursor._query(self,
q)
    308 self._last_executed = q
    309 self._clear_result()
--> 310 conn.query(q)
    311 self._do_get_result()
    312 return self.rowcount

File C:₩ProgramData₩Anaconda3₩lib₩site-packages₩pymysql₩connections.py:548, in Connection.query
(self, sql, unbuffered)
    546     sql = sql.encode(self.encoding, "surrogateescape")
    547 self._execute_command(COMMAND.COM_QUERY, sql)
--> 548 self._affected_rows = self._read_query_result(unbuffered=unbuffered)
    549 return self._affected_rows

File C:₩ProgramData₩Anaconda3₩lib₩site-packages₩pymysql₩connections.py:775, in Connection._read
_query_result(self, unbuffered)
    773 else:
    774     result = MySQLResult(self)
--> 775     result.read()
    776 self._result = result
    777 if result.server_status is not None:

File C:₩ProgramData₩Anaconda3₩lib₩site-packages₩pymysql₩connections.py:1156, in MySQLResult.rea
d(self)
   1154 def read(self):
   1155     try:
-> 1156         first_packet = self.connection._read_packet()
   1158         if first_packet.is_ok_packet():
   1159             self._read_ok_packet(first_packet)

File C:₩ProgramData₩Anaconda3₩lib₩site-packages₩pymysql₩connections.py:725, in Connection._read
_packet(self, packet_type)
    723     if self._result is not None and self._result.unbuffered_active is True:
    724         self._result.unbuffered_active = False
--> 725     packet.raise_for_error()
    726 return packet

File C:₩ProgramData₩Anaconda3₩lib₩site-packages₩pymysql₩protocol.py:221, in MysqlPacket.raise_f
or_error(self)
    219 if DEBUG:
    220     print("errno =", errno)
--> 221 err.raise_mysql_exception(self._data)

File C:₩ProgramData₩Anaconda3₩lib₩site-packages₩pymysql₩err.py:143, in raise_mysql_exception(da
ta)
    141 if errorclass is None:
    142     errorclass = InternalError if errno < 1000 else OperationalError
--> 143 raise errorclass(errno, errval)

IntegrityError: (1451, 'Cannot delete or update a parent row: a foreign key constraint fails (`sqldb
`.`buytbl`, CONSTRAINT `buytbl_ibfk_1` FOREIGN KEY (`userID`) REFERENCES `usertbl` (`userID`))')
```

에러나면 정상입니다.

- buyTbl 에 해당 userID를 참조하는 데이터가 있기 때문

[ ]:

```

```