NTU Database Management System – from SQL to NoSQL – Homework 4

資料科學 R10946013 劉謦瑄

Part 1-1.

```
import pandas as pd
df = pd.read_csv('DBMS_student_list.csv',index_col=0)

# Create DB
from sqlalchemy import create_engine
from sqlalchemy_utils import database_exists, create_database
engine = create_engine("mysql://root:Xuan137986ntu@127.0.0.1/DB_class")
if not database_exists(engine.url):
    create_database(engine.url)
print(database_exists(engine.url))

#Create table
sql = '''create table student(
            身份 CHAR(11) NOT NULL,
            系所 CHAR(11) NOT NULL,
            年級 INT NOT NULL,
            學號 CHAR(11) NOT NULL,
            姓名 CHAR(11)
            );
'''
engine = create_engine("mysql://root:Xuan137986ntu@127.0.0.1/DB_class?charset=utf8")
conn = engine.connect()
try:
    conn.execute(sql)
except Exception:
    print("table already exist!")
engine.connect()

engine = create_engine('mysql://root:Xuan137986ntu@127.0.0.1/DB_class?charset=utf8', echo=False)
#read csv into Dataframe
df = pd.read_csv('DBMS_student_list.csv',index_col=0)
#write csv into Sql
df.to_sql('student', con=engine, if_exists='replace',index=False)
```

```
True

50
```

確認成功建立指定 Database & Table:

```
q = engine.execute('SHOW DATABASES')
available_DB = q.fetchall()
for i in available_DB:
    print(i)
```

```
('books',)
('bookstoredb',)
('DB_class',)
('information_schema',)
('mysql',)
('performance_schema',)
('sys',)
```

```
from sqlalchemy import inspect
insp = inspect(engine)
print(insp.get_table_names())
```

```
['student']
```

```
import sqlalchemy as db
engine = db.create_engine('mysql://root:Xuan137986ntu@127.0.0.1/DB_class?charset=utf8')
connection = engine.connect()
metadata = db.MetaData()
census = db.Table('student', metadata, autoload=True, autoload_with=engine)
# print(census.columns.keys())
query = db.select([census])
ResultProxy = connection.execute(query)
ResultSet = ResultProxy.fetchall()
ResultSet
```

```
Output exceeds the size limit. Open the full output data in a text editor
[('校內生', '土木系結構組', 1, 'r10521219', '丁治鈞'),
 ('校內生', '農藝系生統組', 1, 'r10621203', '何善學'),
 ('校內生', '生醫電資所', 2, 'r09945024', '余銘仁'),
 ('校內生', '電機資安碩班', 1, 'r10921a01', '劉品柄'),
 ('校內生', '資料科學學程', 1, 'r10946013', '劉聲瑄'),
 ('校內生', '電機系', 4, 'b04901126', '卓冠宇')]
```

Part 1-2.

```
import sqlalchemy as db
engine = db.create_engine('mysql://root:Xuan137986ntu@127.0.0.1/DB_class?charset=utf8')
connection = engine.connect()
metadata = db.MetaData()
census = db.Table('student', metadata, autoload=True, autoload_with=engine)
# print(census.columns.keys())
query = db.select([census]).where(census.columns.姓名 == '劉聲瑄')
ResultProxy = connection.execute(query)
ResultSet = ResultProxy.fetchall()
ResultSet
```

```
[('校內生', '資料科學學程', 1, 'r10946013', '劉聲瑄')]
```

Part 1-3.

```
engine = db.create_engine('mysql://root:Xuan137986ntu@127.0.0.1/DB_class?charset=utf8')
connection = engine.connect()
metadata = db.MetaData()
census = db.Table('student', metadata, autoload=True, autoload_with=engine)
query = db.select([census]).where(db.and_(census.columns.系所 == '資料科學學程', census.columns.年級 == 1))
ResultProxy = connection.execute(query)
ResultSet = ResultProxy.fetchall()
ResultSet
```

```
[('校內生', '資料科學學程', 1, 'r10946013', '劉聲瑄'),
 ('校內生', '資料科學學程', 1, 'r10946001', '李奕宏')]
```

Part 1-4.

```
import sqlalchemy as db
import pandas as pd

engine = db.create_engine("mysql://root:Xuan137986ntu@127.0.0.1/DB_class?charset=utf8")
metadata = db.MetaData()
connection = engine.connect()
emp = db.Table('student', metadata, autoload=True, autoload_with=engine)

# Build a statement to update
query = db.update(emp).values(身份 = '特優生')
query = query.where(emp.columns.姓名 == '劉聲瑄')
results = connection.execute(query)
results = connection.execute(db.select([emp])).fetchall()

census = db.Table('student', metadata, autoload=True, autoload_with=engine)
query = db.select([census]).where(census.columns.姓名 == '劉聲瑄')
ResultProxy = connection.execute(query)
ResultSet = ResultProxy.fetchall()
ResultSet
```

```
[('特優生', '資料科學學程', 1, 'r10946013', '劉聲瑄')]
```

Part 1-5.

```
from sqlalchemy import text
sql = '''
    INSERT INTO student
    VALUES
        ('旁聽生','歷史系',1,'b09900201','小花'),
        ('校內生','歷史系',4,'b06900332','小草'),
        ('校內生','機械系',4,'b06502055','小天');
'''
with engine.connect().execution_options(autocommit=True) as conn:
    conn.execute(text(sql))
```

Part 1-6.

```
engine = db.create_engine('mysql://root:Xuan137986ntu@127.0.0.1/DB_class?charset=utf8')
connection = engine.connect()
metadata = db.MetaData()

census = db.Table('student', metadata, autoload=True, autoload_with=engine)
for i in ['b09900201','b06900332','b06502055']:
# print(census.columns.keys())
    query = db.select([census]).where(census.columns.學號 == i)
    ResultProxy = connection.execute(query)
    ResultSet = ResultProxy.fetchall()
    print(ResultSet)
```

```
[('旁聽生', '歷史系', 1, 'b09900201', '小花')]
[('校內生', '歷史系', 4, 'b06900332', '小草')]
[('校內生', '機械系', 4, 'b06502055', '小天')]
```

Part 2-1. & 2-2.

```
/* Part2 - 1&2*/
PREPARE Part2_1 FROM
'SELECT ISBN, type, Price, Author_ID, Editor_ID FROM Book
WHERE Price NOT Between 0 AND 300 AND (Editor_ID=1 OR Author_ID=?)';

SET @Author_ID = 1;
EXECUTE Part2_1 USING @Author_ID;
```

```
Query OK, 0 rows affected (0.00 sec)

+------------+--------------+-------+-----------+-----------+
| ISBN       | type         | Price | Author_ID | Editor_ID |
+------------+--------------+-------+-----------+-----------+
| 1234567890 | 一般圖書     |  1000 |         1 |         1 |
+------------+--------------+-------+-----------+-----------+
1 row in set (0.00 sec)
```

Part 2-3.

```
/* Part2 - 3*/

DELIMITER $$
CREATE FUNCTION discount(Price INT) RETURNS INT
DETERMINISTIC
BEGIN
    DECLARE new INT;
    SET new = Price*0.8;
    RETURN new;
END$$
DELIMITER ;

SELECT ISBN,type, Price, discount(Price)
FROM Book
ORDER BY type;
```

```
Query OK, 0 rows affected (0.01 sec)

+------------+--------------+-------+------------------+
| ISBN       | type         | Price | discount(Price) |
+------------+--------------+-------+------------------+
| 1034128394 | 一般圖書     |   200 |              160 |
| 1234567890 | 一般圖書     |  1000 |              800 |
| 1107893846 | 兒童圖書     |   400 |              320 |
| 1356782901 | 成人書籍     |   100 |               80 |
+------------+--------------+-------+------------------+
4 rows in set (0.00 sec)
```

Part 2-4.

```sql
/* Part2 - 4*/

DELIMITER $$
CREATE PROCEDURE DB_class.get_Count(IN depat CHAR(11), OUT STCOUNT INT)
BEGIN
    SELECT COUNT(*) FROM student where 系所 = depat;
END$$
DELIMITER ;

CALL DB_class.get_Count('資料科學學程' ,@STCOUNT);
CALL DB_class.get_Count('電機系' ,@STCOUNT);
DROP PROCEDURE IF EXISTS DB_class.get_Count;
```

```
Query OK, 0 rows affected (0.00 sec)

+----------+
| COUNT(*) |
+----------+
|        3 |
+----------+
1 row in set (0.00 sec)
```

&

```
Query OK, 0 rows affected (0.00 sec)

+----------+
| COUNT(*) |
+----------+
|       10 |
+----------+
1 row in set (0.00 sec)
```

Part 2-5.

```sql
/* Part2 - 5*/

-- CREATE TABLE before_insert(Price INT);
DELIMITER $$
Create Trigger Item_Amount_Total
BEFORE INSERT ON Library_Item FOR EACH ROW
BEGIN
    SET @SUM = @SUM+New.Avalibale;
END$$
DELIMITER ;

SET @SUM = 0;
SELECT @SUM AS 'Amount BEFORE YOU INSERT';
SELECT * From Library_Item;

INSERT INTO Library_Item VALUES(14, '第四本雜誌', 1,1,Null);
INSERT INTO Library_Item VALUES(15, '第五本雜誌', 1,2,Null);
SELECT @SUM AS 'Amount AFTER YOU INSERT';
SELECT * From Library_Item;
```

```
Query OK, 0 rows affected (0.00 sec)

+----------------------+
| Amount BEFORE INSERT |
+----------------------+
|                    0 |
+----------------------+
1 row in set (0.00 sec)

+----+-----------------+-----------+----------------------+-----------------+
| ID | Title           | Avalibale | Fk_PublishingOrg_Id | Fk_Borrower_Id |
+----+-----------------+-----------+----------------------+-----------------+
|  1 | 好看的書        |         1 |                    1 |            NULL |
|  2 | 更好看的書      |         0 |                    2 |               1 |
|  3 | 最好看的書      |         0 |                    3 |               1 |
|  4 | 好好看的書      |         0 |                    2 |               2 |
|  5 | 第一本雜誌      |         1 |                    2 |            NULL |
|  6 | 第二本雜誌      |         0 |                    1 |               2 |
|  7 | 第三本雜誌      |         1 |                    3 |            NULL |
|  8 | Journal第一集   |         0 |                    1 |               1 |
|  9 | Journal第二集   |         0 |                    2 |               3 |
| 10 | Journal第三集   |         1 |                    1 |            NULL |
+----+-----------------+-----------+----------------------+-----------------+
10 rows in set (0.00 sec)

Query OK, 1 row affected (0.00 sec)

+---------------------+
| Amount AFTER INSERT |
+---------------------+
|                   2 |
+---------------------+
1 row in set (0.00 sec)

+----+-----------------+-----------+----------------------+-----------------+
| ID | Title           | Avalibale | Fk_PublishingOrg_Id | Fk_Borrower_Id |
+----+-----------------+-----------+----------------------+-----------------+
|  1 | 好看的書        |         1 |                    1 |            NULL |
|  2 | 更好看的書      |         0 |                    2 |               1 |
|  3 | 最好看的書      |         0 |                    3 |               1 |
|  4 | 好好看的書      |         0 |                    2 |               2 |
|  5 | 第一本雜誌      |         1 |                    2 |            NULL |
|  6 | 第二本雜誌      |         0 |                    1 |               2 |
|  7 | 第三本雜誌      |         1 |                    3 |            NULL |
|  8 | Journal第一集   |         0 |                    1 |               1 |
|  9 | Journal第二集   |         0 |                    2 |               3 |
| 10 | Journal第三集   |         1 |                    1 |            NULL |
| 14 | 第四本雜誌      |         1 |                    1 |            NULL |
| 15 | 第五本雜誌      |         1 |                    2 |            NULL |
+----+-----------------+-----------+----------------------+-----------------+
```

Part 2-6.

```sql
/* Part2 - 6*/
DELIMITER //

create table LibraryItem_audit_Create (ID integer, created_by varchar (30));
CREATE TRIGGER LibraryItem_after_insert
AFTER INSERT ON Library_Item FOR EACH ROW
BEGIN
   DECLARE vUser varchar(50);
   -- Find username of person performing the INSERT into table
   SELECT USER() INTO vUser;
   -- Insert record into audit table
   INSERT INTO LibraryItem_audit_Create
   ( ID,
     created_by)
   VALUES
   ( NEW.ID,
     vUser );
END; //

DELIMITER ;

insert into Library_Item values (11, 'Journal第四集', 1,1,Null);
insert into Library_Item values (12, 'Journal第五集', 1,1,Null);
insert into Library_Item values (13, 'Journal第六集', 1,1,Null);
select *from LibraryItem_audit_Create;
```

```
DELIMITER //

create table LibraryItem_audit_Delete (ID integer, deleted_by varchar(30));
CREATE TRIGGER LibraryItem_after_delete
AFTER DELETE ON Library_Item FOR EACH ROW
BEGIN
   DECLARE vUser varchar(50);
   -- Find username of person performing the DELETE into table
   SELECT USER() INTO vUser;
   -- Insert record into audit table
   INSERT INTO LibraryItem_audit_Delete
   ( ID,
     deleted_by)
   VALUES
   ( OLD.ID,
     vUser );
END; //

DELIMITER ;

delete from Library_Item WHERE Library_Item.ID = 11;
delete from Library_Item WHERE Library_Item.ID = 12;
select *from LibraryItem_audit_Delete;
```

```
Query OK, 1 row affected (0.00 sec)

+------+----------------+
| ID   | created_by     |
+------+----------------+
|   11 | root@localhost |
|   12 | root@localhost |
|   13 | root@localhost |
+------+----------------+
3 rows in set (0.01 sec)

Query OK, 1 row affected (0.01 sec)

+------+----------------+
| ID   | deleted_by     |
+------+----------------+
|   11 | root@localhost |
|   12 | root@localhost |
+------+----------------+
2 rows in set (0.00 sec)
```