[3-2. 데이터베이스로부터 데이터 읽기]

1. Connector 모듈의 connection 객체 소개

1-1. connection 객체 개요

```
connection = mysql.connector.connect(host="localhost",
user="root", ...)
```

- connection객체는 mysql.connector.connect() 함수를 사용하여 데이터베이스에 연결하고 반환된 객체로, MySQL 데이터베이스에 대한 연결을 나타냅니다.
- 연결에 필요한 정보(사용자 이름, 비밀번호, 호스트, 데이터베이스 등)를 포함합니다.
- 데이터베이스와의 커넥션을 설정하고 유지하는 역할을 합니다.
- 데이터베이스 연결을 사용한 작업이 끝나면 "close()" 메서드를 호출하여 연결을 닫아야 합니다.

1-2. connection 객체 주요 속성

- connection_id : 연결의 서버 ID를 반환합니다. 서버가 연결을 식별하는 데 사용됩니다.
- user : 연결에 사용된 MySQL 사용자 이름을 반환합니다.
- server_host : 연결된 MySQL 서버의 호스트 이름을 반환합니다.
- server_port : 연결된 MySQL 서버의 포트 번호를 반환합니다.
- database : 연결된 데이터베이스 이름을 반환합니다.

In [ ]:
```python
from connector import Connector # Connector 모듈 import

with Connector() as conn:
    connection = conn.connection # connection 객체
    print(f"connection_id : {connection.connection_id}")
    print(f"user          : {connection.user}")
    print(f"server_host   : {connection.server_host}")
    print(f"server_port   : {connection.server_port}")
    print(f"database      : {connection.database}")
```

```
connection_id : 257
user          : root
server_host   : localhost
server_port   : 3306
database      : cashflow
```

1-3. connection 객체 주요 메서드

- cursor() : 데이터베이스의 쿼리 실행을 위해 커서 객체를 생성합니다. 커서는 SQL 명령을 실행하고, 결과를 조회하는 데 사용됩니다.
- commit() : 트랜잭션 내에서 실행된 모든 변경사항을 데이터베이스에 확정합니다.
- rollback() : 현재 트랜잭션에서 실행된 모든 변경사항을 취소합니다.
- close() : 데이터베이스 연결을 닫습니다. 이 메서드를 호출하면 더 이상 해당 연결 객체를 사용할 수 없습니다.
- is_connected() : 현재 데이터베이스 서버와의 연결 상태를 반환합니다. 연결이 활성 상태라면 True, 그렇지 않다면 False를 반환합니다.

> 트랜잭션 : 하나 이상의 쿼리들을 하나의 작업 단위로 묶는 것을 의미. 이 작업 단위는 모두 함께 성공적으로 실행되거나, 하나라도 실패할 경우 전체가 취소(롤백)되어야 함. 이를 통해 데이터의 일관성과 무결성을 유지할 수 있음.

In [ ]:
```python
from connector import Connector # Connector 모듈 import

with Connector() as conn:
    connection = conn.connection # connection 객체
    cursor_new = connection.cursor() # cursor 객체 생성 후 cursor_new 변수에
    print(f"cursor_new    : {cursor_new}")
    print(f"is_connected  : {connection.is_connected()}")
```

```
cursor_new    : CMySQLCursor: (Nothing executed yet)
is_connected  : True
```

2. Connector 모듈의 cursor 객체 소개

2-1. cursor 객체 개요

```
cursor = connection.cursor()
```

- cursor객체는 connection.cursor() 메서드를 호출하여 생성되며, 쿼리를 실행하고 그 결과를 처리하는 데 사용됩니다.
- 데이터베이스와 상호 작용하기 위한 메서드와 속성을 제공합니다. 예를 들어, 'execute()' 메서드를 사용하여 쿼리를 실행하고, 'fetchall()' 메서드를 사용하여 실행 결과를 가져올 수 있습니다.
- 쿼리의 실행, 결과 처리, 데이터 가져오기 등의 역할을 합니다.
- 모든 작업이 완료되면 "close()" 메서드를 호출하여 커서를 닫아야 합니다.

2-2. cursor() 메서드 주요 인자

- dictionary
  - 기본값 : False
  - 이 옵션을 True로 설정하면, 커서는 쿼리 결과를 딕셔너리 형태로 반환합니다. 이는 컬럼명을 Key로 사용하여 결과에 접근할 수 있게 하여 코드의 가독성을 높여줍니다.
- buffered
  - 기본값 : False
  - 이 옵션을 True로 설정하면, 커서가 쿼리의 모든 결과를 즉시 서버로부터 가져와 내부 버퍼에 저장합니다. 이는 여러 행을 처리할 때 유용하지만, 많은 양의 데이터를 다룰 때 메모리 사용량이 증가할 수 있습니다.

2-2-1. dictionary가 False인 경우

In [ ]:
```
import mysql.connector

connection = mysql.connector.connect(
    host="localhost",
    user="root",
    port="3306",
    password="fastcampus1!",
    database="cashflow"
)
cursor = connection.cursor()

query = "SELECT * FROM accounts_info"
cursor.execute(query)
result = cursor.fetchall()
print(result)

# 연결 종료
cursor.close()
connection.close()
```

```
[(1, 'hana1234', '하나은행', '12345-01-1234', 'FC호텔', '보통예금', '여의도점', '매출계좌', '매출대금 입금계좌', datetime.date(2024, 1, 2), None, datetime.datetime(2024, 4, 10, 23, 33, 35), None), (2, 'hana4321', '하나은행', '12345-01-4321', 'FC호텔', '보통예금', '여의도점', '운영계좌', '운영비용 지출 계좌', datetime.date(2024, 1, 2), None, datetime.datetime(2024, 4, 10, 23, 33, 35), None), (3, 'shhn4567', '신한은행', '54321-21-4567', 'FC호텔', '정기예금', '여의도점', '정기예금', '10억원 1년 정기예금', datetime.date(2024, 1, 2), None, datetime.datetime(2024, 4, 10, 23, 33, 35), None)]
```

In [ ]:
```
type(result)
```

Out[ ]:
```
list
```

In [ ]:
```
result[0]
```

Out[ ]:
```
(1,
 'hana1234',
 '하나은행',
 '12345-01-1234',
 'FC호텔',
 '보통예금',
 '여의도점',
 '매출계좌',
 '매출대금 입금계좌',
 datetime.date(2024, 1, 2),
 None,
 datetime.datetime(2024, 4, 10, 23, 33, 35),
 None)
```

In [ ]:
```
import pandas as pd
pd.DataFrame(result)
```

Out[ ]:

	0	1	2	3	4	5	6	7	8	9	10	11	12
0	1	hana1234	하나은행	12345-01-1234	FC호텔	보통예금	여의도점	매출계좌	매출대금 입금계좌	2024-01-02	None	2024-04-10 23:33:35	None
1	2	hana4321	하나은행	12345-01-4321	FC호텔	보통예금	여의도점	운영계좌	운영비용 지출계좌	2024-01-02	None	2024-04-10 23:33:35	None
2	3	shhn4567	신한은행	54321-21-4567	FC호텔	정기예금	여의도점	정기예금	10억원 1년 정기예금	2024-01-02	None	2024-04-10 23:33:35	None

2-2-2. dictionary가 True인 경우

In [ ]:
```python
import mysql.connector

connection = mysql.connector.connect(
    host="localhost",
    user="root",
    port="3306",
    password="fastcampus1!",
    database="cashflow"
)
cursor = connection.cursor(dictionary=True)

query = "SELECT * FROM accounts_info"
cursor.execute(query)
result2 = cursor.fetchall()
print(result2)

# 연결 종료
cursor.close()
connection.close()
```

```
[{'id': 1, 'account_id': 'hana1234', 'bank_name': '하나은행', 'account_numb
er': '12345-01-1234', 'account_holder': 'FC호텔', 'account_type': '보통예
금', 'branch_name': '여의도점', 'account_alias': '매출계좌', 'description': '
매출대금 입금계좌', 'opening_date': datetime.date(2024, 1, 2), 'closing_dat
e': None, 'created_at': datetime.datetime(2024, 4, 10, 23, 33, 35), 'del
eted_at': None}, {'id': 2, 'account_id': 'hana4321', 'bank_name': '하나은
행', 'account_number': '12345-01-4321', 'account_holder': 'FC호텔', 'accou
nt_type': '보통예금', 'branch_name': '여의도점', 'account_alias': '운영계좌', '
description': '운영비용 지출 계좌', 'opening_date': datetime.date(2024, 1,
2), 'closing_date': None, 'created_at': datetime.datetime(2024, 4, 10, 2
3, 33, 35), 'deleted_at': None}, {'id': 3, 'account_id': 'shhn4567', 'ba
nk_name': '신한은행', 'account_number': '54321-21-4567', 'account_holder':
'FC호텔', 'account_type': '정기예금', 'branch_name': '여의도점', 'account_alia
s': '정기예금', 'description': '10억원 1년 정기예금', 'opening_date': datetime.
date(2024, 1, 2), 'closing_date': None, 'created_at': datetime.datetime(
2024, 4, 10, 23, 33, 35), 'deleted_at': None}]
```

In [ ]:
```python
result2[0]
```

Out[ ]:
```
{'id': 1,
 'account_id': 'hana1234',
 'bank_name': '하나은행',
 'account_number': '12345-01-1234',
 'account_holder': 'FC호텔',
 'account_type': '보통예금',
 'branch_name': '여의도점',
 'account_alias': '매출계좌',
 'description': '매출대금 입금계좌',
 'opening_date': datetime.date(2024, 1, 2),
 'closing_date': None,
 'created_at': datetime.datetime(2024, 4, 10, 23, 33, 35),
 'deleted_at': None}
```

In [ ]:
```
pd.DataFrame(result2)
```

Out[ ]:

	id	account_id	bank_name	account_number	account_holder	account_type	branch_
0	1	hana1234	하나은행	12345-01-1234	FC호텔	보통예금	0
1	2	hana4321	하나은행	12345-01-4321	FC호텔	보통예금	0
2	3	shhn4567	신한은행	54321-21-4567	FC호텔	정기예금	0

3. SELECT 문을 이용하여 데이터 읽어오기

3-1. Connector 모듈 import 하기

In [ ]:
```
import pandas as pd
from connector import Connector
```

3-2. SELECT 이용하여 데이터 추출하기

3-2-1. 입금액이 0보다 큰 데이터 전체 추출하기

In [ ]:
```
with Connector() as conn:
    query = "SELECT * FROM cashflow_tbl WHERE deposit > 0;"
    conn.cursor.execute(query)
    result = pd.DataFrame(conn.cursor.fetchall())
```

In [ ]:
```
result
```

Out[ ]:

	id	account_id	dw_date	category	client	deposit	withdrawal	description	cr
0	1	hana4321	2024-01-02	자본금	FC사	2000000000	0	자본금 입금	2
1	3	shhn4567	2024-01-02	계좌이체	FC사	1000000000	0	정기예금 예치	2
2	4	hana1234	2024-01-03	객실수입	호텔스닷컴	33000000	0	객실 매출	2
3	7	hana1234	2024-01-04	객실수입	호텔스넷	25000000	0	객실 매출	2
			2024-		호텔스				2

4	8	hana1234	01-05	객실수입	닷컴	30000000	0	객실 매출	
5	9	hana1234	2024-01-06	객실수입	호텔스넷	25000000	0	객실 매출	2
6	10	hana1234	2024-01-07	객실수입	호텔스닷컴	31000000	0	객실 매출	2
7	12	hana1234	2024-01-08	객실수입	호텔스넷	25000000	0	객실 매출	2
8	14	hana1234	2024-01-09	객실수입	호텔스닷컴	31000000	0	객실 매출	2
9	15	hana1234	2024-01-10	객실수입	호텔스넷	27000000	0	객실 매출	2
10	19	hana1234	2024-01-11	객실수입	호텔스닷컴	25000000	0	객실 매출	2
11	20	hana1234	2024-01-12	객실수입	호텔스넷	32000000	0	객실 매출	2
12	21	hana1234	2024-01-13	객실수입	호텔스닷컴	35000000	0	객실 매출	2
13	23	hana1234	2024-01-14	객실수입	호텔스넷	33000000	0	객실 매출	2
14	25	hana1234	2024-01-15	객실수입	호텔스닷컴	24000000	0	객실 매출	2
15	26	hana1234	2024-01-16	객실수입	호텔스넷	22000000	0	객실 매출	2
16	27	hana1234	2024-01-17	객실수입	호텔스닷컴	25000000	0	객실 매출	2
17	30	hana1234	2024-01-18	객실수입	호텔스넷	28000000	0	객실 매출	2
18	31	hana1234	2024-01-19	객실수입	호텔스닷컴	25000000	0	객실 매출	2
19	34	hana1234	2024-01-20	객실수입	호텔스넷	27000000	0	객실 매출	2
			2024-		호텔스				2

20	36	hana1234	01-21	객실수입	닷컴	29000000	0	객실 매출	
21	38	hana1234	2024-01-22	객실수입	호텔스넷	28000000	0	객실 매출	2
22	39	hana1234	2024-01-23	객실수입	호텔스닷컴	31000000	0	객실 매출	2
23	40	hana1234	2024-01-24	객실수입	호텔스넷	25000000	0	객실 매출	2
24	44	hana1234	2024-01-25	객실수입	호텔스닷컴	30000000	0	객실 매출	2
25	46	hana1234	2024-01-26	객실수입	호텔스넷	31000000	0	객실 매출	2
26	48	hana1234	2024-01-27	객실수입	호텔스닷컴	25000000	0	객실 매출	2
27	49	hana1234	2024-01-28	객실수입	호텔스넷	30000000	0	객실 매출	2
28	50	hana1234	2024-01-29	객실수입	호텔스닷컴	32000000	0	객실 매출	2
29	51	hana1234	2024-01-30	객실수입	호텔스넷	27000000	0	객실 매출	2
30	53	hana1234	2024-01-31	객실수입	호텔스닷컴	32000000	0	객실 매출	2

3-2-2. 각 계좌별 전체 입금/출금 합계액 및 계좌잔고 구하기

In [ ]:
```
with Connector() as conn:
    query = """
        SELECT
            account_id,
            SUM(deposit) AS total_deposit,
            SUM(withdrawal) AS total_withdrawal,
            SUM(deposit) - SUM(withdrawal) AS balance
        FROM cashflow_tbl
        GROUP BY account_id;
    """
    conn.cursor.execute(query)
    result = pd.DataFrame(conn.cursor.fetchall())
```

In [ ]:
```
result = result.set_index("account_id")
result
```

Out[ ]:

account_id	total_deposit	total_withdrawal	balance
hana4321	2000000000	1680150000	319850000
shhn4567	1000000000	0	1000000000
hana1234	823000000	0	823000000

In [ ]:
```
result.loc['hana4321', 'balance']
```

Out[ ]:
```
Decimal('319850000')
```

In [ ]:
```

```