Background

- ✓ CRUD
- ✓ SELECT & JOIN
- ✓ DAO

Goal

- ✓ JDBC API 를 사용하여 DB 연결 및 Query 처리한다..
- ✓ 등록, 수정, 삭제 외에 다양한 SELECT 를 처리한다.
- ✓ DB Access 관련 처리를 DAO 클래스로 구현한다.

Problem

ITSKB 은행은 새로운 고객 관리 시스템을 구축하고 있다. DB 와 연동하기 위해 JDBC 를 적용하기로 하였다. account 테이블에 대한 CRUD 프로그램을 작성하고 users 테이블과 함께 다양한 SELECT 문을 작성한다. 또한 주요 SQLException 오류를 확인하고 해결해 보자.

- 1. eclipse 에 프로젝트를 새로 구성한다. (프로젝트명 : DB_WS_05)
- 2. 프로젝트 코드에서 JDBC 로 Oracle DB 에 접속할 수 있도록 Oracle JDBC Driver 를 프로젝트에 추가한다.
- 3. jdbc.dao 패키지를 만들고 아래의 파일을 추가한다.
 - interface : AccountDao,java
 - class : AccountDaoImpl.java
 - Test.java 에 필요한 DB Access 기능을 구현한다.
 - 구현 class 는 Singleton 디자인 패턴을 적용한다.

🦒 DB 5. JDBC 응용

- 4. jdbc.dto 패키지를 만들고 아래의 class 을 추가한다.
 - UserDto.java : DB_WS_04 와 동일
 - AccountDto.java
 - UserAccountCntDto.java
- ** 아래의 패키지 구성을 참고한다.
 - > 🔃 Test.java jdbc.common > DBManager.java AccountDao.java
 - > AccountDto.java
 - J UserAccountCntDto.java
 - > UserDto.java
- 5. Test.java 코드를 완성한다.
 - 1 번 ~ 8 번까지 순차적으로 코드를 완성해 가면서 DB 테이블의 결과도 함께 확인한다.
 - 1 번 ~ 5 번은 account 테이블에 대한 기본적인 CRUD 기능을 코드로 작성한다.
 - 6 번 ~ 8 번은 account 및 users 테이블에 대한 join 등을 통해 구현해 본다.
- ** Test.java 코드의 comment 를 참고하여 필요한 기능을 구현해 본다.

6. 실행되는 화면 예시는 다음과 같다.

```
insert()
```

```
select * from account where account_seq = 80;
```

		⊕ BALANCE	∯ USER_SEQ
80	800808080800800	8000	333

update()

```
select * from account where account_seq = 80;
```

ACCOUNTLSEQ		⊕ BALANCE	⊕ USER_SEQ
80	00800808008008	5000	333

selectAll() eclipse - console

```
AccountDto [accountSeq=50, accountNumber=00500505050505, balance=3000, userSeq=222, name=null]
AccountDto [accountSeq=10, accountNumber=0010010100101, balance=1000, userSeq=111, name=null]
AccountDto [accountSeq=30, accountNumber=00300303003003, balance=5000, userSeq=222, name=null]
AccountDto [accountSeq=70, accountNumber=00700707007007, balance=6500, userSeq=444, name=null]
AccountDto [accountSeq=40, accountNumber=0040040400404, balance=4500, userSeq=222, name=null]
AccountDto [accountSeq=60, accountNumber=00600606060606, balance=2000, userSeq=222, name=null]
AccountDto [accountSeq=80, accountNumber=008008080808, balance=5000, userSeq=333, name=null]
AccountDto [accountSeq=20, accountNumber=00200202002002, balance=6000, userSeq=111, name=null]
```

selectOne() eclipse - console

```
AccountDto [accountSeq=80, accountNumber=0080080808080, balance=5000, userSeq=333, name=null]
```

delete()

```
select * from account where account_seq = 80;
```

ACCOUNT_SEQ | ⊕ ACCOUNT_NUMBER | ⊕ BALANCE | ⊕ USER_SEQ

selectBalanceDescTop3() eclipse - console

```
AccountDto [accountSeq=70, accountNumber=007007070070, balance=6500, userSeq=444, name=null] AccountDto [accountSeq=20, accountNumber=00200202002002, balance=6000, userSeq=111, name=null] AccountDto [accountSeq=30, accountNumber=00300303003003, balance=5000, userSeq=222, name=null]
```

selectUsersAccountByName() eclipse - console

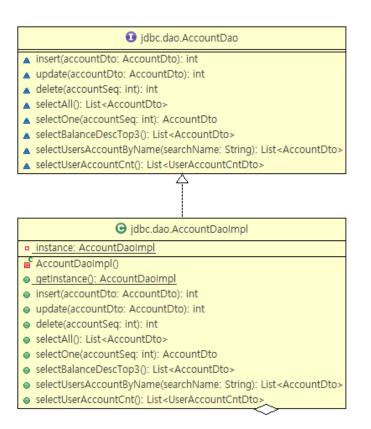
```
AccountDto [accountSeq=0, accountNumber=00500505005005, balance=3000, userSeq=222, name=이길동] AccountDto [accountSeq=0, accountNumber=00100101001001, balance=1000, userSeq=111, name=홍길동] AccountDto [accountSeq=0, accountNumber=00300303003003, balance=5000, userSeq=222, name=이길동] AccountDto [accountSeq=0, accountNumber=00700707007007, balance=6500, userSeq=444, name=사길동] AccountDto [accountSeq=0, accountNumber=00400404004004, balance=4500, userSeq=222, name=이길동] AccountDto [accountSeq=0, accountNumber=0060060606006, balance=2000, userSeq=222, name=이길동] AccountDto [accountSeq=0, accountNumber=002002020020020, balance=6000, userSeq=111, name=홍길동]
```

selectUserAccountCnt() eclipse - console

```
UserAccountCntDto [userSeq=222, name=이길동, accountCnt=4] UserAccountCntDto [userSeq=444, name=사길동, accountCnt=1] UserAccountCntDto [userSeq=111, name=홍길동, accountCnt=2] UserAccountCntDto [userSeq=555, name=오길동, accountCnt=0] UserAccountCntDto [userSeq=333, name=삼길동, accountCnt=0]
```



7. 작성하는 코드는 아래의 클래스 다이어그램을 참고한다.



⊙ jdbc.app.Test			
0	main(args: String[]): void		
Δ,	insert(): void		
À,	update(): void		
Ä,	selectAll(): void		
Δ	selectOne(): void		
Δ	delete(): void		
A	selectBalanceDescTop3(): void		
A	selectUsersAccountByName(): void		
A	selectUserAccountCnt(): void		

⊙ jdbc.dto.UserDto

- userSeq: int
- name: String
- email: String
- phone: String
- isSleep: boolean
- getUserSeq(): int
- setUserSeq(userSeq: int): void
- getName(): String
- setName(name: String): void
- getEmail(): String
- setEmail(email: String): void
- getPhone(): String
- setPhone(phone: String): void
- isSleep(): boolean
- setSleep(isSleep: boolean): void
- toString(): String

⊙ jdbc.dto.UserAccountCntDto

- userSeq: int
- name: String
- accountCnt: int
- accountent. Int
- getUserSeq(): int
- setUserSeq(userSeq: int): void
- getName(): String
- setName(name: String): void
- getAccountCnt(): int
- setAccountCnt(accountCnt: int): void
- toString(): String

jdbc.dto.AccountDto

- accountSeq: int
- accountNumber: String
- balance: int
- userSeq: int
- name: String
- getAccountSeq(): int
- setAccountSeq(accountSeq: int): void
- getAccountNumber(): String
- setAccountNumber(accountNumber: String): void
- getBalance(): int
- setBalance(balance: int): void
- getUserSeq(): int
- setUserSeq(userSeq: int): void
- getName(): String
- setName(name: String): void
- toString(): String

❖ 제출방법

- 프로젝트명: "DB_WS_05" 으로 작성
- 완성 후 프로젝트를 DB_WS_05_반_성명.zip으로 압축하여 제출