<데이터베이스 실습 복습 정리>

161205 to 161207 (JDBC)

우리가 흔히 쓰는 프로그래밍 언어가 Java가 되겠다. 그래서 우리는 나중에 자바 프로그래밍에서도 데이터베이스를 다룰 수 있는 JDBC의 원리에 대해서 공부를 해 볼 필요가 있다. 필자도 JDBC에 대해서도 요약본을 정리해 줌으로서 다 같이 공부하면서 다 같이 가는 마음으로 정리하였으니 열심히 공부해줬으면 좋겠다.

[요약 중점 사항]

- JDBC의 원리에 대해 숙달한다.
- DAO와 Statement의 개념에 대해 알아둔다.
- 실제로 써먹는 JDBC 코딩 방법에 대해서 정리하면서 어떤 방식으로 쓸지에 대해 Select문부터 Delete문까지를 통해서 살펴보도록 한다.

---

final. JDBC at 161206

이제 데이터베이스 실습 요약정리도 이것으로 마지막으로 끝낸다. 이번에 공부해 보는 것은 원래 고급 웹 프로그래밍1에서 배우게 되지만, 데이터베이스 실습 시간에 선행학습 한다고 생각을 하는 마음에서 공부를 하는 것이 좋겠다. 이번에는 마지막으로 필자가 작성한 코딩과 함께 요약정리를 보내줄테니 기말고사에 좋은 성과가 있길 바라면서 바로 들어가보겠다.

1. JDBC

우리가 흔히 쓰는 SQL 언어를 자바나 파이썬, 심지어 C언어까지 연동이 되는 개념이 있다. 자바를 사례로 들어서 우리가 흔히 쓰는 데이터베이스 SQL문에 대해서 자바에서 구동을 할 수 있는 개념이 바로 Java API(Application Programing Inferface)라고 칭한다. JDBC는 어떤 특성이 있는지 간략히 살펴보자.

-> JDBC 드라이버는 오라클만이 있는 것이 아니다. 예를 들어 오라클 SQL뿐만이 아니라 MySQL 등등에 대해 연동을 할 수 있다. 여기서 SQL는 다른 회사에서 만들었어도 기본적인 개념들은 공통점을 발견할 수 있지만, 함수나 프로시저 등의 사정은... 우선 기말고사부터 어떻게 하자.

-> 개발자가 JDBC API를 사용하면 DBMS에 적당한 JDBC 드라이버(ojdbc6_g.jar 블라블라한거)만 있으면 어떤 DB를 다루든 이용이 가능하다.

-> 물론 JDBC 드라이버는 각 밴드 사에서 jar 파일로 제공을 한다. 우리가 흔히 쓰는 회사인 오라클로 예시를 든다면, 오라클과 미국에 있는 DB 소프트웨어 회사 인포믹스 등등의 각종 회사가 SQL 질의에 맞게 자바 프로그래밍으로 구현이 되어 있어서 JTBC 드라이버를 이용해서 응용프로그램을 클라이언트에서 실행하는 원리로 생각하면 되겠다.

2. DAO 객체

카트라이더의 그 캐릭터를 생각하는 사람이 있겠다만, 여기서는 Databace Access Object를 줄여서 쓴 말이다. 이는 데이터베이스의 데이터를 select하고, 심심하면 insert하고, 꼴리면 Update하고, 뭐같음 Delete하는 개체로 볼 수 있다.

-> 일반적으로 DAO와 DB테이블에 대해서 1vs1 혹은 1vs多로 매핑을 한다. 그렇지만 우리가 알아볼 경우는 1vs1이니 후반의 경우는 심심할 때 구글링을 통해서 미리 고웹1에 대해 예습하는 것을 추천한다.

3) JDBC 순서

JDBC는 잘만 쓰면 약이 되고, 필자처럼 사용자 변경만 잘해주면 되는데 우리가 쓰는 호스트 이름은 흔하게 localhost를 이용할 수 있다. 그렇지만 우리가 여태껏 실습실에서 SID를 xe로 써서 JDBC가 실행이 되지 않는 경우가 더러 있어서 필자가 고생했었다. 그렇지만 SID 중에 orcl이라고 있는데 이거로 수정을 해보면 정상 작동이 된다. 집에서 여러분들이 SQL Developer를 이용할 때 SID를 무엇으로 사용하는지에 대하여 참조해서 properties 파일을 잘 살펴보길 바란다. 또한 이번에는 코딩을 통해서 여러분들이 쉽게 접근하기 위해 161206_JDBC_문장으로_알아보자.hwp를 통해서 필자의 설명도 보충해서 추가했으니 이에 대해서 참조하면 도움이 되겠다.

3-1) JDBC 드라이버 로딩

우선 우리가 프로젝트에 우측 마우스를 눌러서 Build Path를 통해서 ojdbc6_g.jar 파일을 불러왔을 것이다. 이러한 파일이 바로 JDBC 드라이버 파일이 되어서 그 파일을 통해서 JDBC를 실행하는 과정이라고 생각하면 되겠다. 물론 버전마다 다른 버전들이 있는데 우리는 6_g를 주로 쓸 것이다.

3-2) 데이터베이스 Connection 연결

우리가 데이터베이스에 대해서 필요로 한 것이 무엇일까? 바로 연결(Connection)이다. 그래서 우리가 주로 JDBC에서 DAO를 형성한 클래스에서 Select문, Delete문, Update문 등을 실행할 때 처음에 해야 하는 작업이 바로 Connection인데 이는 우리가 쓰는 파일에서 util에 있는 DB의 클래스 메소드인 getConnecion()을 통해서 얻어오면 되겠다.

3-3) 쿼리 문장을 실행하기 위한 Statement 객체 형성

Statement는 우리가 자바에서 SQL 문장을 실행하기 위해서 필요로 한 객체이다. Statement는 PreparedStatement(준비된 Statement), CallableStatement(호출하는 Statement)로 구성이 된다. 우리가 무난하게 Select, Insert, Update, Delete 문 등을 통해서 Java에서 연동을 할 수 있다면 PreparedStatement를 이용하면 되는데, PL/SQL에서 배웠던 프로시저, 함수, 패키지 등을 불러올 때에는 CallableStatement를 이용하게 된다.

3-4) 쿼리 실행

우리가 방금 생성한 PreparedStatement 객체를 이용해서 연결 객체를 통해서 prepareStatement라는 메소드를 통해서 SQL 문장을 올리는 역할을 한다. 그렇지만 SQL 문장을 왜 여기에다가 올리는가에 대해서 궁금증이 있을 것이다. 앞에서 Statement의 역할에 대해서 설명을 안 했지만 이는 우리가 작성한 쿼리(SQL) 문장들을 접속 개체에 올려서 연결 객체를 통해서 쿼리 문장을 실행해주는 역할을 해준다.

3-5) 쿼리 실행에 있어서 결과를 저장하는 과정

쿼리를 작성하고 Statement 객체에 올리면 무슨 과정이 필요로 할까? 바로 ResultSet라는 java.sql 패키지에 있는 결과 값 저장 전용 컬렉션을 이용하면 된다. 이를 통해서 PreparedStatement에서 실행한 메소드(executeQuery())를 처리하고 난 뒤에는 분명 결과 값들이 나오게 된다. 그래서 ResultSet 객체를

이용해 다음과 같이 저장을 하게 된다면 쿼리들의 결과물이 이쪽으로 짬때려지게 된다. 만일 매개변수가 Primary Key라면 하나의 결과만 나와서 result의 각 결과 노드에 대해 존재하는지에 대한 next() 메소드를 통해서 결과가 존재하면 makeJob(ResultSet result) 메소드를 통해서 형성을 하면 되겠다.

3-6) 사용한 객체 종료
커서와 마찬가지로 각 개체들을 사용하고 난 뒤에는 정리를 잘 해줘야 한다. 이는 close 메소드를 이용하면 되는데 ResultSet, Statement 관련 객체, Connection 객체들을 닫아주도록 실행하면 되겠다.

이처럼 6단계의 과정을 거쳐야지 JDBC를 잘 활용할 수 있도록 정리를 또 다른 파일에 정리해뒀으니 과정을 살펴보면서 같이 공부하면 도움 되겠다.

4) Properties 파일

Properties 파일은 쉽게 이야기해서 데이터베이스를 연결하는데 필요로 한 파일 중의 하나라고 보면 된다. 이의 구성은 4가지로 쉽게 나뉘어서 볼 수 있는데 어떻게 구성이 되어 있는지에 대해 정리해볼 필요가 있겠다.
JDBC_DRIVER_NAME : 우리가 어떠한 드라이버를 이용해서 JDBC를 연동하는지에 대해 작성을 하는 정보라고 보면 되겠다. 이는 데이터베이스 회사에 따라 오라클도 있을 수 있고, mySQL도 될 수가 있다.
ex) oracle.jdbc.driver.OracleDriver
DB_URL : 데이터베이스를 연결해주는 주소이다. thin 옆에 골뱅이부터 살펴보면 localhost는 우리가 연결 할 호스트 이름으로서, IP주소도 되고 그 주소에 대한 키워드로 작성이 가능하다. 1521는 포트로 보면 되고, orcl는 SID로서 orcl, xe로 나뉠 수 있으니 집에서 여러분들이 연결할 때 쓰는 SID에 대해 구분을 해서 작성하면 되겠다.
ex) jdbc:oracle:thin:@localhost:1521:orcl
USER_ID : 말 그대로 User 이름이다. User 이름은 SYS, HR, Scott 등등이 있다.
USER_PASSWORD : 말 그대로 비밀번호이다. 늘 그래왔듯이 우리가 흔히 쓰는 6202로 설정해두면 되겠다.

5) JDBC 내에서 쓰이는 Transaction

우리가 오라클을 공부하면서 값들을 변경하거나 추가, 삭제하는 경우에는 SQL Developer에서는 commit을 바로 안 해줘서 실수했을 경우에 rollback을 하거나 중간 지점을 만드는데 savepoint를 이용할 수 있었다. 그렇지만 JDBC에서는 DML문장을 작성하게 되면 애석하게도 바로 바로 Commit이 되기 때문에 자동적으로 Commit이 되지 않게 설정을 해야 한다.
-> connect.setAutoCommit(false)
이는 Connection에 있는 메소드 중에 하나인데, 원래 기본 설정 값인 디폴트는 true로 되어 있어서 바로 즉시 저장을 하게 된다. 마찬가지로 예외가 발생하게 되면 자동적으로 rollback도 된다. 그렇지만 오라클처럼 작업들에 대해서 되살펴보면서 하고 싶다면 이를 false로 설정해주고 작업을 계속하면 rollback을 통해서 실수한 부분에 대해서 제어할 수 있으며, 완수되었다면 commit를 하면 되겠다.
또한 JDBC 내에서 쓰는 트렌젝션은 Connection 클레스에 있는 메소드로 살펴보면 되는데 대표적으로 우리가 쓰는 commit(), rollback(), 심지어 저장점을 설정하는 setSavepoint("저장점메소드")도 있으니 이는 참고로만 알아두면 되겠다.