1. 핵심 모듈 및 알고리즘 설명
   1. **SQLBoolean Class  
      삼진 논리(three-valued logic)를 구현하여 TRUE, FALSE, UNKNOWN 값을 처리**
   2. **\_check\_where\_clause()  
      주어진 record가 where절 조건에 맞는지 확인하는 함수. delete, select등에서 사용**
   3. **\_join\_tables()  
      여러 개의 테이블을 join하는 함수. select의 from절에서 사용**
   4. **\_select\_columns()  
      select된 column들만 return하는 함수. select에서 사용**
   5. **\_perform\_order\_by()  
      order by구문을 따라 \_join\_tables()의 결과로 나온 records를 재정렬함**
   6. **\_print\_select()  
      select의 최종 결과를 출력**
2. 구현 내용 요약  
   SQL기반의 구문들(INSERT, DELETE, SELECT)을 처리할 수 있는 데이터베이스를 구현함.
   1. INSERT  
      입력된 값의 데이터 타입을 검증하고, 스키마에 맞게 데이터를 저장  
      NOT NULL 제약 조건만을 확인. primary key의 uniqueness나 foreign key는 확인 안 함
   2. DELETE  
      WHERE 절 조건에 맞는 데이터를 삭제. 외부에서 참조되고있는 table은 truncate때와 마찬가지로 삭제 불가.
   3. SELECT  
      WHERE 절과 ORDER BY 절을 처리하며, SELECT \* 또는 특정 컬럼을 선택적으로 출력. JOIN을 지원하여 여러 테이블의 데이터를 결합할 수 있음.
   4. Group by  
      group by 구문의 일부를 구현함.  
      group by 동작 시 record의 값이 null인 경우 처리 못함.  
      group by 동작 시 aggregate의 결과 값이 하나로 합쳐져 나오지 않고 그냥 여러 번 출력됨.
3. 느낀 점  
   **SQL의 다양한 기능을 구현하면서 데이터베이스의 내부 동작을 깊이 이해할 수 있었다. 특히, JOIN 알고리즘을 구현하는 과정에서 SQL의 복잡성을 체감했다.**
4. 기타사항
   1. 구현 사항  
      이 과제에서 나의 구현은 다음과 같음.  
      1. 출력 시 column\_name앞에는 항상 table\_name이 붙음.  
      2. join시 공유되는 column을 하나로 합치지 않고 따로따로 둘 다 출력.  
      3. 하지만 2의 구현으로 인해 동일한 이름의 column을 join으로 합치고 그 column\_name을 사용하는 경우 ambiguous하다고 봄.
   2. 추가 구현  
      1. insert에서 명시적 column입력의 순서가 schema와 달라도 정상적으로 작동함.  
      2. order by 구문에서도 table\_name.column\_name의 입력이 가능.  
      3. order by 구문에서 ASC/DESC가 없는 경우 ASC로 정렬됨.
   3. 추가로 정의된 오류  
      1. insert에서 명시적 column입력에서 column이름이 중복되어 들어오는 경우   
      에러 발생, “column name is duplicated” 출력  
      2. join에서 잘못된 join\_condition이 들어오는 경우  
      에러 발생, “Wrong join condition” 출력  
      3. order by 구문에 입력된 table\_name이 from절에 없는 table인 경우   
      에러 발생, TableNotSpecified(#clauseName) 출력

.