

ICE4016 데이터베이스설계

<Week12 Practice>

보고서 작성 서약서

1. 나는 타학생의 보고서를 베끼거나 여러 보고서의 내용을 짜집기하지 않겠습니다.

2. 나는 보고서의 주요 내용을 인터넷사이트 등을 통해 얻지 않겠습니다.

3. 나는 보고서의 내용을 조작하지 않겠습니다.

4. 나는 보고서 작성에 참고한 문헌의 출처를 밝히겠습니다.

5. 나는 나의 보고서를 제출 전에 타학생에게 보여주지 않겠습니다.

나는 보고서 작성시 윤리에 어긋난 행동을 하지 않고 정보통신공학인으로서 나의 명예를 지킬 것을 맹세합니다.

2023년 11 월 22 일

학부 정보통신공학

학년 4

성명 이재영

학번 12181816

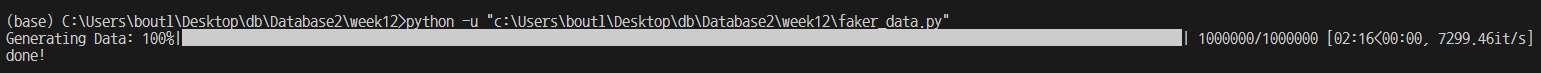
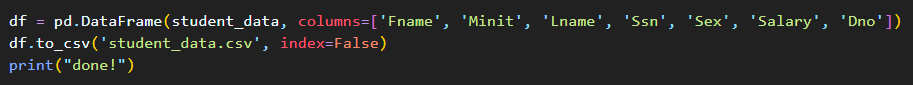


1. 개요

* 10만명 이상의 학생 데이터를 Inha\_DB에 INSERT하는 쿼리 작성
* SHOW INDEX의 cardinality, EXPLAIN의 rows, filtered 조사
* 10만명 이상의 학생 데이터를 삽입한 뒤 5개 이상의 INDEX 적용 전후 비교 및 분석

1. 상세 설계 내용

* 텍스트, 스크린샷, 폰트이(가) 표시된 사진

  자동 생성된 설명100만명의 학생 데이터 생성
  + 파이썬에서 faker 모듈을 활용해 100만명의 학생 데이터를 생성
    - first\_name(): fname을 무작위 생성하는 부분
    - pystr(max\_chars=1).upper(): Minit을 무작위 생성하는 부분, 한 자리의 str(알파벳)을 무작위로 생성하며, 조교님께서 올려 주신 예제 쿼리에는 모두 대문자여서 upper()로 대문자 변환
    - last\_name(): Lname을 무작위 생성하는 부분
    - pyint(min\_value=100000000, max\_value=999999999): Ssn을 무작위 생성하는 부분, min\_value ~ max\_value 범위 중 무작위로 정수를 생성
    - random\_element(elements=(‘M’,’F’)): Sex를 무작위 생성하는 부분, ‘M’,’F’중 무작위로 선택
    - pyint(min\_value=10000, max\_value=99999): Salary를 무작위로 생성하는 부분
    - pyint(min\_value=1, max\_value=9): Dno를 무작위로 생성하는 부분
  + tdqm 라이브러리를 사용해 데이터 생성 진행상황 및 소요 시간 확인
  + 생성된 데이터를 Pandas 라이브러리로 csv 파일로 로컬에 저장
  + 텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

    자동 생성된 설명텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

    자동 생성된 설명student\_data.csv
    - 100만 개의 튜플이 생성되어 올바르게 저장된 것을 확인
* MySQL 접속 시 로컬 파일 로드 기능 OFF임을 확인 후 활성화
  + 텍스트, 폰트, 스크린샷이(가) 표시된 사진

    자동 생성된 설명해당 기능 OFF 되어 있음
  + 활성화
  + MySQL 재접속 (--local\_infile 부분 추가)
* 텍스트, 스크린샷, 폰트이(가) 표시된 사진

  자동 생성된 설명데이터 INSERT 쿼리
  + ‘inha\_week12’ 데이터 베이스와 Student 테이블을 생성하고 로컬 파일 업로드 기능을 활용하여 열의 속성을 나타내는 첫 번째줄을 제외한 student\_data.csv 상의 100만개의 튜플을 Student 테이블에 INSERT
  + 텍스트, 폰트, 스크린샷이(가) 표시된 사진

    자동 생성된 설명100만 개의 튜플이 삽입되는데 약 12초가 걸리는 것을 확인할 수 있음.
  + 텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

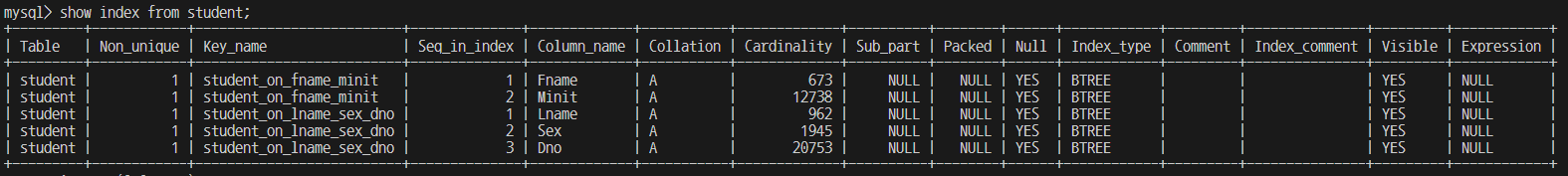
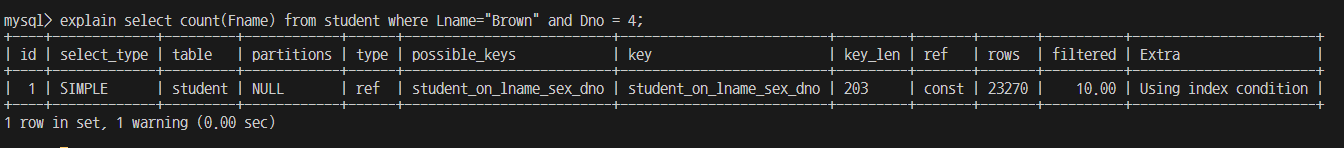
    자동 생성된 설명텍스트, 스크린샷이(가) 표시된 사진

    자동 생성된 설명Student 테이블 정보 및 데이터 확인

1. 실행 화면

* 텍스트, 폰트, 스크린샷, 라인이(가) 표시된 사진

  자동 생성된 설명스크린샷, 텍스트, 폰트, 라인이(가) 표시된 사진

  자동 생성된 설명CREATE INDEX
* SHOW INDEX
  + 해당 쿼리 시 Cardinality 항목을 확인할 수 있는데 이는 테이블 전체에서 얼마나 중복되는 지를 보여주는 지표라고 생각할 수 있다. Minit, Dno는 다른 attribute에 비해 Cardinality가 큰 것을 볼 수 있는데 Minit은 A~Z 범위의 한자리 알파벳, Dno는 1~9 범위의 한자리 정수 이기 때문에 중복도가 높은 것을 유추할 수 있다.
* EXPLAIN
  + rows
    - 해당 항목은 explain 뒤 select문, 즉 원하는 튜플을 찾기 위해 얼마나 많은 튜플에 접근해야 할지에 대한 예측 개수 값을 의미, 통계 값으로 예측된 값이기 때문에 실제 접근 튜플 수와 일치하지 않을 수도 있음.
  + Filtered
    - 총 튜플 수를 rows의 예상 튜플 수라 하면, where문의 조건에 따라 걸러지게 되면 남게되는 튜플의 비율을 의미, rows: 23270, filtered:10.00라면 23270개의 튜플 중 where문의 조건에 부합하여 남게 되는 튜플의 비율이 10%라는 것이다. 하지만 이 값도 예측된 값이기 때문에 정확하지 않을 수 있다.
* INDEX 적용 전후 비교
  + 텍스트, 스크린샷, 폰트, 소프트웨어이(가) 표시된 사진

    자동 생성된 설명INDEX 적용 후
    - 이미 INDEX를 적용했기 때문에 먼저 적용 후의 select 쿼리 실행
  + 텍스트, 폰트, 스크린샷이(가) 표시된 사진

    자동 생성된 설명INDEX 적용 전
    - 텍스트, 스크린샷, 폰트이(가) 표시된 사진

      자동 생성된 설명DROP INDEX 문을 이용해 인덱스 삭제
    - select문을 실행했을 때 현저히 느려진 것을 확인할 수 있었음.
  + select count(Fname) from student where Lname=”Lee”; 문에서 INDEX 적용 시 (1.15 – 0.08) / 1.15 \* 100 = 약 93%의 성능 개선
  + select count(Fname) from student where Lname=”Park”; 문에서 INDEX 적용 시 (1.38 – 0.01) / 1.38 \* 100 = 약 99%의 성능 개선
  + select count(Fname) from student where Lname= “Brown”; 문에서 INDEX 적용 시 (1.12 -0.08) / 1.12 \* 100 = 약 93%의 성능 개선

🡺 INDEX 적용 시에 뛰어난 성능 개선율을 보이는 것을 확인할 수 있음.

1. 결론

* 이번 과제를 이용해서 파이썬 라이브러리를 이용해 대용량 데이터를 생성해보고, 대용량 데이터를 빠르게 데이터 베이스에 INSERT하는 코드를 활용할 수 있었습니다.
* MySQL의 INDEX를 적용하여 대용량 데이터에서 빠르게 select문이 동작하는 것을 확인할 수 있었으며 Term\_project에서 데이터베이스 테스트를 할 때 활용해보도록 노력하겠습니다.