2024 Spring Database

# PROJECT 1-2 INTRODUCTION

# Project 1-2

- Implementing DDL & Basic DML functions
  - Project 1-1에 기반하여 시작
  - **[스키마를 저장하고 관리]**할 수 있는 기능을 구현
    - create table …
    - drop table …
    - explain/describe/desc …
    - show tables
  - **[스키마에 데이터를 추가하고 검색]**할 수 있는 기능 일부 구현
    - insert
    - select
  - **1-1에서 구현한 파서와, Berkeley DB**를 활용해서 구현

# Project 1-2 목표

① 스키마를 저장/삭제 (CREATE, DROP)

② 저장된 스키마 조회 (EXPLAIN/DESC/DESCRIBE/SHOW)

③ 스키마에 데이터 추가 (INSERT)
  ① 간단한 INSERT 구문 구현 (INSERT INTO *table_name)*
  ② INSERT 구문 구현은 1-3에서 완성 예정

④ 스키마에 저장된 데이터 조회 (SELECT)
  ① 전체를 SELECT하는 기능 구현 (SELECT * FROM *table_name*)
  ② SELECT 구문 구현은 1-3에서 완성 예정

# Berkeley DB

- What is Berkeley DB?
  - DBMS를 구현할 수 있도록 API를 제공하는 라이브러리
  - Provided by Oracle
    - 임베디드 DB software를 제공하는 것이 초기 개발 목적
    - 라이브러리형 DB로 확장성과 성능이 장점
  - <key, value>를 byte array 형태로 저장
  - C++, Java, Python 등 다양한 언어의 API 지원
  - Not a relational system
    - 스키마 설계, 데이터 관리 방법은 여러분들의 몫
    - ‘One DB-One Schema’, ‘One DB-Multi Schema’, etc.

# Berkeley DB

- References
  - (Oracle) https://docs.oracle.com/cd/E17076_05/html/programmer_reference/index.html
  - (Python Version & API Resources)

    https://www.jcea.es/programacion/pybsddb.htm

    https://docs.jcea.es/berkeleydb/latest/index.html

# Berkeley DB

- Installation
  - `pip install berkeleydb`
  - (Optional – Ubuntu의 경우) 위 `pip install` 에서 에러가 발생하는 경우
    아래 명령어 실행 후 재시도.

    ```
    sudo apt-get install libdb-dev
    ```
  - (Optional-Mac의 경우) 위 `pip install` 에서 에러가 발생하는 경우, homebrew로 requirements를 설치하고 난 뒤, BerkeleyDB path를 명시해주어야 함.

    ```
    brew install berkeley-db@5
    BERKELEYDB_DIR=$(brew --prefix berkeley-db@5) pip install
    berkeleydb
    ```
  - 필요할 시 신청하시면 서버공간을 제공해 드릴 수는 있습니다.

# 파싱 결과물 활용 예시

```python
class MyTransformer(Transformer):
    def __init__(self, prompt):
        super().__init__()
        self.prompt = prompt

    def create_table_query(self, items):
        # Fetch elements from the tree generated by parsing the query
        table_name = items[2].children[0].lower()
        column_definition_iter = items[3].find_data("column_definition")
```

- items
  - 입력 받은 쿼리에 대한 파싱 결과물 (1-1에서 설명한 Tree 구조 참고)
- find_data(str)
  - 파싱된 tree에서 str을 이름으로 하는 토큰을 root node로 하는 subtree의 nodes를 iterable object로 반환
- References
  - https://lark-parser.readthedocs.io/en/latest/visitors.html#transformer
  - https://lark-parser.readthedocs.io/en/latest/classes.html#lark.Tree

# Berkeley DB – Create, Open, Close

## Open with Creation

```
from berkeleydb import db
myDB = db.DB()
myDB.open('myDB.db', dbtype=db.DB_HASH, flags=db.DB_CREATE)
myDB.close()
```

## Just Open

```
from berkeleydb import db
myDB = db.DB()
myDB.open('myDB.db', dbtype=db.DB_HASH)
myDB.close()
```

# Berkeley DB – Put, Get, Iterate

### Put, Get

```
myDB.put(b"apple", b"red")
print(myDB.get(b"apple"))  # b"red"
```

### Iterate All

```
cursor = myDB.cursor()
while x := cursor.next():
    print(x)  # (key, value)
```

※ 이 외의 기능들을 사용해도 됩니다!
※ 다만, Berkeley DB 11g부터 SQL이 지원되며, SQLite 라이브러리와 호환이 됩니다.
**따라서 이에 해당되는 API 사용을 금지합니다.**
Reference: https://www.oracle.com/technical-resources/articles/database/oracle-berkeley-db-sql-api.html
※ 위 예제 코드는 python3.8이상에서 작성되었습니다. (walrus operator ":=")

# QnA

- Q. DB 파일은 어디에 저장되어야 하나요?
  - DB 파일의 경로는 **반드시** 프로젝트 폴더 내부에 저장되어야 합니다.

  예) 프로젝트 폴더명: PRJ1-2_2024-12345
  DB 파일 경로: PRJ1-2_2024-12345/myDB.db

  - **복수의 DB** 파일로 저장해야 하는 경우, 프로젝트 폴더 내부에 '**DB**'라는 이름의 폴더를 만들어 거기에 저장하셔야 합니다.

  예) 프로젝트 폴더명: PRJ1-2_2024-12345

  DB 파일 경로:
  PRJ1-2_2024-12345/DB/myDB1.db
  PRJ1-2_2024-12345/DB/myDB2.db
  …

- Q. 프로그램이 종료되면 DB 파일이 어떻게 되어야 하나요?
  - 여러분들의 스키마와 데이터는 파일에 저장되어야 합니다. 즉, 종료되어도 DB파일은 최근 상태를 유지하고 있어야 합니다.

# QnA

- Q. DB 파일을 같이 제출해야 하나요?
  - DB 파일을 제출하실 필요는 없습니다. 올바른 경로에 DB파일이 생성, 저장되기만 하면 됩니다.
  - 필요한 경우, 빈 DB 파일을 같이 제출해도 됩니다.
    - 즉, DB를 초기화하고 제출하셔야 됩니다.
    - DB가 초기화된 상태라고 가정하고 채점이 진행됩니다.
    - DB에 데이터가 들어있는 상태로 제출하면 채점에서 불이익을 받으실 수 있습니다.

# Notice

- **제출 기한: 2024/4/23(일) 11:59pm**
  - **10% penalty for ~24 hours late**
  - **20% penalty for 24~48 hours late**
  - **no credit after 48 hours**

- 자세한 설명은 스펙 문서를 참고
- 문서와 파일 포맷을 다시 한번 확인하고 제출
- **프로젝트 전반에 관한 문의는 ETL PRJ1-2 Q&A 게시판을 통해 4/19(금)까지 (**비밀글 게시 및 이메일 문의는 지양)
- **개별 코드 구현에 대한 문의는 형평성에 따라 불가합니다.**