SDS, Uni Passau
Winter term 2020/21
miniHive, Milestone 1

The goal of the coding project is to build a mini-version of Apache Hive, called *miniHive*. The first milestone is to write a query compiler that translates simple SQL queries into relational algebra. Later, we will add selection pushing to optimize the query, and then we will compile the relational algebra query into a physical query plan of MapReduce jobs. But you are not completely on your own with this...

Translating SQL into Relational Algebra

We consider SQL statements that are conjunctive queries, of the form

SELECT DISTINCT $A_1, \ldots, A_n$
FROM T_1 $t_1, \ldots,$ $T_m t_m$
WHERE C

where

- $A_1, \ldots, A_n$ are attribute names,
- $T_1, \ldots, T_m$ are relation names,
- $t_1, \ldots, t_n$ are optional renamings,
- and C is a conjunction of equality conditions of the form $t_i.A = t_j.B$ or $t_i.A = c$, where c is a constant, and A and B are attribute names.

The first step in *miniHive* is to translate SQL statements into relational algebra. We make use of two existing Python modules:

- We use `sqlparse` to parse SQL statements.

  More on this module at `https://github.com/andialbrecht/sqlparse`.

- We use `radb` to handle relational algebra statements.

  More on this module at `https://github.com/junyang/radb`.

Write a module `sql2ra` that takes a parsed SQL statement and performs the canonical translation into relational algebra, using the operators σ, π, $\times$, and ρ. Use Python 3.8 (Praktomat uses Python 3.8.5, specifically) for your implementation.

This is how it should work when you spin up the interactive Python interpreter:

```
>>>import sqlparse
>>>import radb
>>>import sql2ra
>>>
>>> sql = "select name from person where gender='female'"
>>> stmt = sqlparse.parse(sql)[0]
>>>
>>> ra = sql2ra.translate(stmt)
>>>
>>> type(ra) # Important! Do not return a raw String, but a Select object.
<class 'radb.ast.Select'>
>>>
>>> print(ra)
\project_{name} (\select_{gender = 'female'} person)
```

Remarks: For Milestone 1, you are not asked to find any particular optimizations to make your implementation more efficient. All you are required to provide is a correct and clean implementation.

Praktomat will use the unit tests of `test_sql2ra.py` to check your solution. Your solution should also work for other, similar queries as in `test_sql2ra.py`. Make sure you upload `sql2ra.py` as a *single* file in Praktomat, in time for the deadline.

When registering at Praktomat, please make sure you use your proper first- and lastname, the same as in Stud.IP. Also use your correct matriculation number and your Uni Passau mail address for the registration. Note that you must be in the Uni Passau VPN to access Praktomat:

`praktomat.sdbs.fim.uni-passau.de`

Your browser may show you the following or a similar warning: "This website is not safe". This is due to a self-signed certificate, but can be ignored.

There will be a plagiarism check. It worked really well in last year's course.
