WPI

Implementing Recursive Queries on Hive and Hadoop

Shijing Yang and Davis Catherman

Background

Apache Hive:

- Implemented on top of Hadoop
- SQL like language
- Does not support recursion natively
  - Unlike SQL, Apache Spark, Javascript, Python, etc

Problem Statement

How can recursion be implemented on Hive in an efficient and logical manner that feels as if it is natively supported while working with different tables and data.

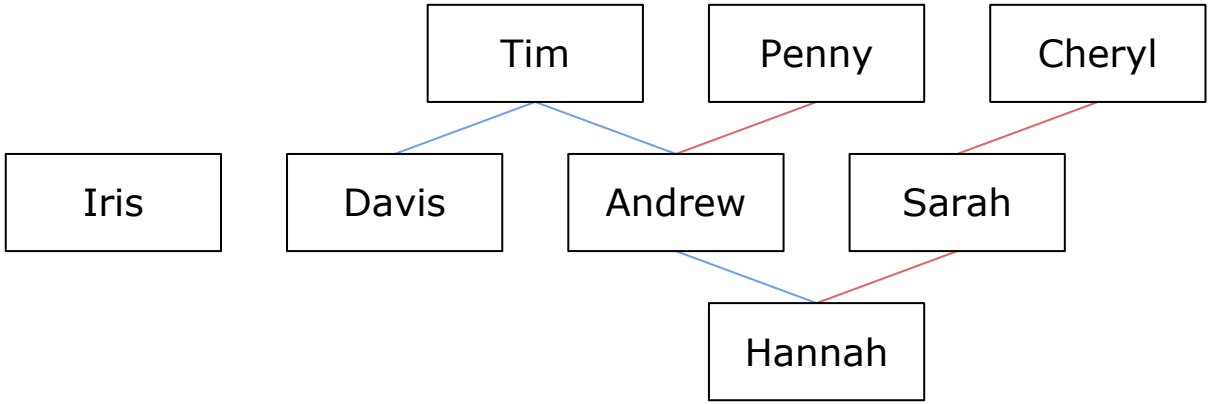
Importance

- Recursion is a useful tool
  - The `Books category` example for MongoDB’
- Switching databases is often impractical
  - All of the data to move
- Switching languages is bad practice
  - Harder to maintain

Importance - Example

Who are the fraternal ancestors of Hannah?

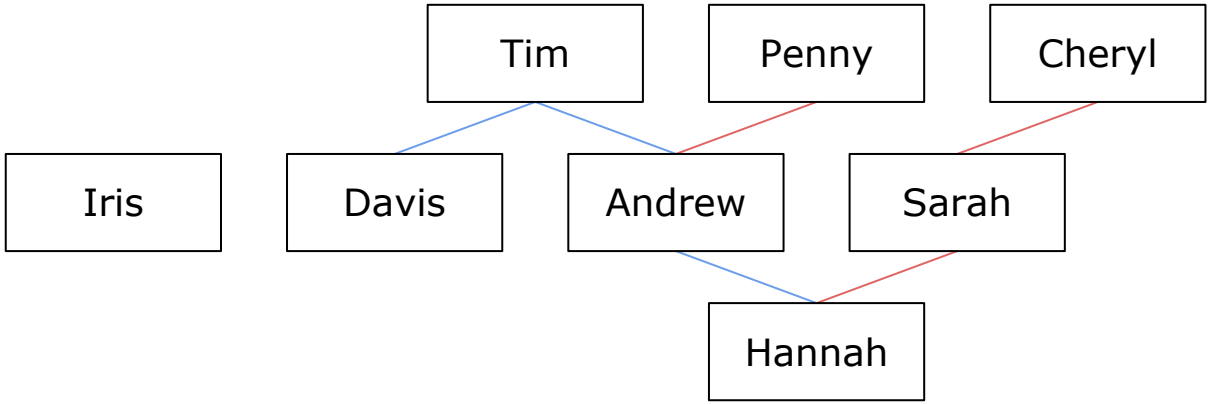
ID	Name	Father	Mother
1	Davis	6	7
2	Iris	null	null
3	Andrew	6	7
4	Sarah	null	8
5	Hannah	3	4
6	Tim	null	null
7	Penny	null	null
8	Cheryl	null	null



Importance - Example

Who are all of the ancestors of Hannah?

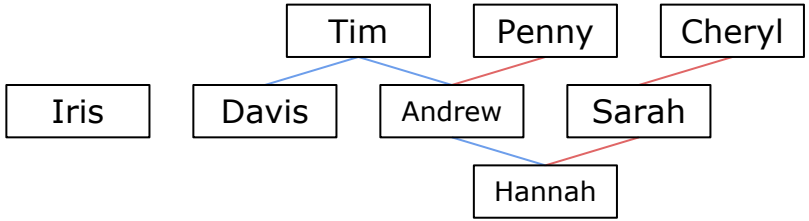
ID	Name	Father	Mother
1	Davis	6	7
2	Iris	null	null
3	Andrew	6	7
4	Sarah	null	8
5	Hannah	3	4
6	Tim	null	null
7	Penny	null	null
8	Cheryl	null	null



Naive Solutions

- Switch database or language
- Define a new syntax style from different language
- User in the loop
  a. Write the recursive query in SQL
  b. Parse/Generate a command for the n-th iteration
  c. Wait for the user to run the command
  d. User checks if complete
  e. Repeat from b.

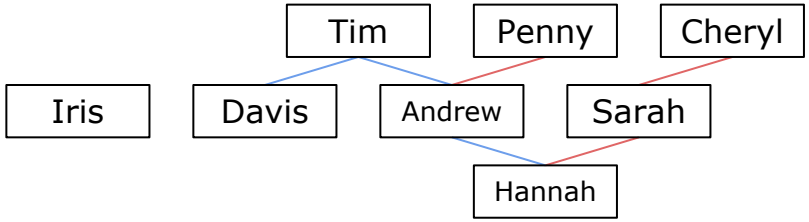
ID	Name	Father	Mother
1	Davis	6	7
2	Iris	null	null
3	Andrew	6	7
4	Sarah	null	8
5	Hannah	3	4
6	Tim	null	null
7	Penny	null	null
8	Cheryl	null	null



Better Solution

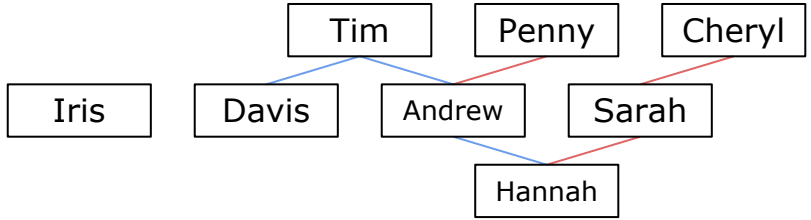
- Use a language to connect to Hive
- Automated
  a. Write the recursive query in SQL
  b. Parse/Generate a command for the n-th iteration
  c. Execute the parsed command, wait for complete
  d. Perform check to see if complete
  e. If not, repeat from b.

ID	Name	Father	Mother
1	Davis	6	7
2	Iris	null	null
3	Andrew	6	7
4	Sarah	null	8
5	Hannah	3	4
6	Tim	null	null
7	Penny	null	null
8	Cheryl	null	null



Recursion in SQL

```
WITH RECURSIVE cte_name AS (
    initial_query  -- anchor member
    UNION ALL
    recursive_query -- recursive member that references to the CTE name
)
SELECT * FROM cte_name;
```



Who are all of the ancestors of Hannah?

In SQL syntax

```
with recursive ancestor as (
select * from people where Name = "Hannah"

union all

select p.* from people p inner join ancestor a on a.father = p.ID or a.mother = p.ID
)

select * from ancestor;
```

Base case

5	Hannah	3	4

Join with the original database

3	Andrew	6	7
4	Sarah	null	8

https://www.mysqltutorial.org/mysql-recursive-cte/

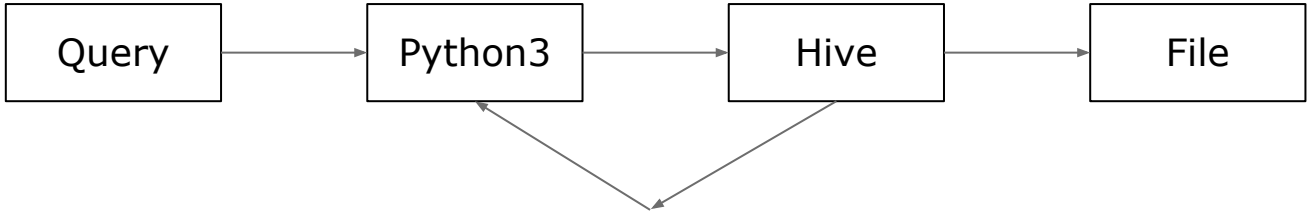
Recursion in Hive

Based off SQL command:

```
'with recursive ancestor as (select * from people where Name = "Hannah" union all select
people.* from people inner join ancestor on ancestor.father = people.ID or
ancestor.mother = people.ID) select * from ancestor;'
```

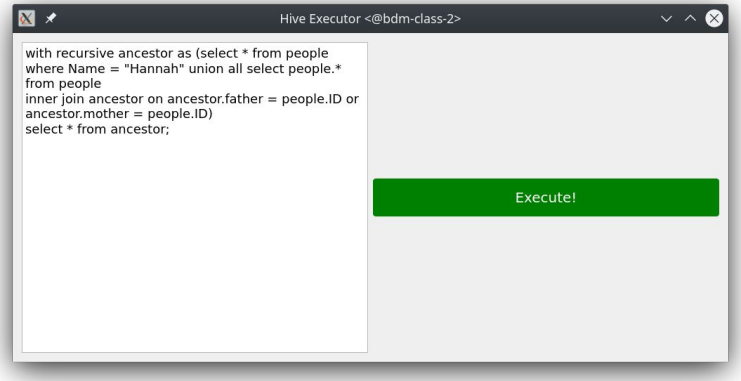
1. Process base case and output to a file
2. Load the output file and create a temporary table
3. Process recursive query
4. Repeat 2 and 3 until the output file is empty
5. Concatenate all previous output
6. Process the final query

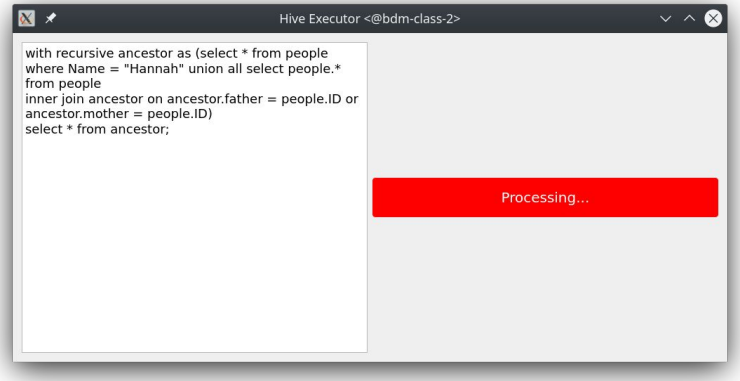
Architecture



Interfaces

- Execute via Command line
- Execute via GUI
  - Made with PyQt5

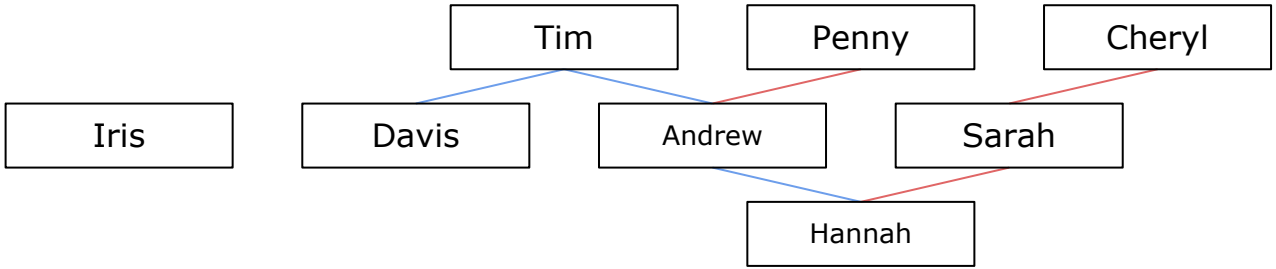




Variations of Query

```
with recursive ancestor as (select * from people where Name = "Hannah" union all select people.* from people inner join ancestor on ancestor.father = people.ID or ancestor.mother = people.ID) select * from ancestor;
```

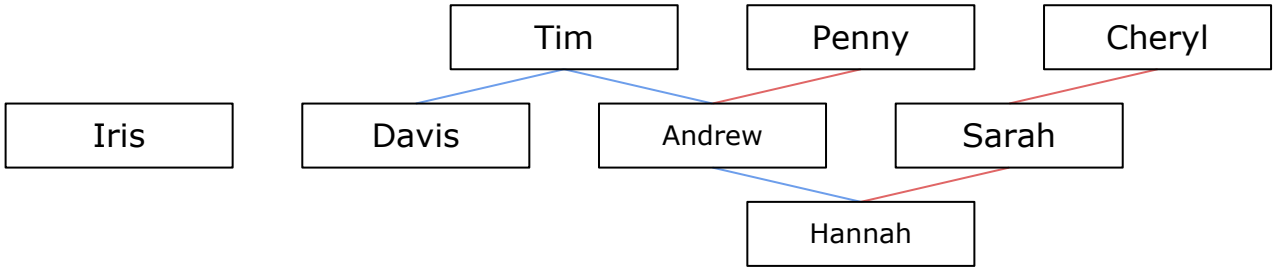
```
with recursive ancestor as (select * from people where Name = "Davis" union all select people.* from people inner join ancestor on ancestor.father = people.ID) select * from ancestor;
```



Variations of Query

```
with recursive ancestor as (select * from people where Name = "Hannah" union all select
people.* from people inner join ancestor on ancestor.father = people.ID or
ancestor.mother = people.ID) select * from ancestor;
```

```
with recursive ancestor as (select * from people where Name = "Hannah" union all select
people.* from people inner join ancestor on ancestor.father = people.ID or
ancestor.father+1) select Name from ancestor;
```



Another Example - finding consecutive numbers

num
8
10
11
12
14

Table description

A table with only one column

Objective

Finding numbers that are greater than and continuous of a selected number

Example

10 → 11,12

8 → null

Implementation in SQL

Input

```
with recursive margin as
(select * from numbers where num = 11
union all
select numbers.* from numbers
inner join margin on margin.num + 1 = numbers.num)
select * from margin;
```

num
8
10
11
12
14

Output

11
12

Conclusion & Future Work

- We are able to perform recursion in Hive!
- Unique vs. with the help of Spark
- Larger benefit with more nodes
- Add terminal to GUI to support

- Demo!

Demo

Questions?