DBMS II
Final Project Presentation

2020 Spring - CS 562
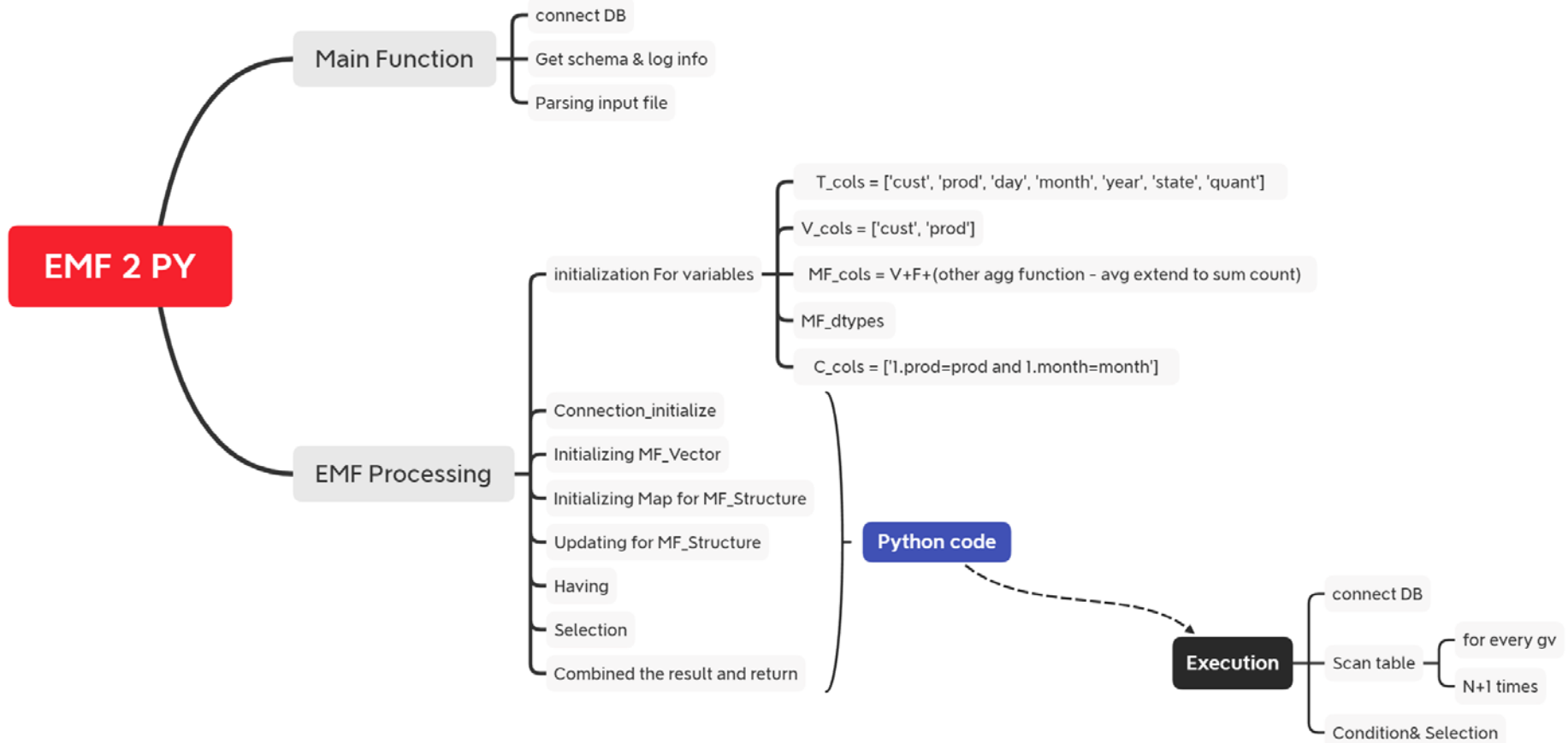
Group: Inscriber

Wei Yang

Why EMF?

- Succinct

- Easy to maintain

- Easy to handle and understand for Non-technical

- Potential to improve the efficiency when embed algorithm

- Convertible with Standard Query

Outline

- Workflow

- Query Structure

- Technology Description

- Limitations

- Demo

- Recap and Forward Looking

Workflow

Query Structure – sample 2

- **SELECT ATTRIBUTE(S):**

  - prod,month, x_sum_quant, y_sum_quant

- **NUMBER OF GROUPING VARIABLES(n):**

  - x, y

- **GROUPING ATTRIBUTES(V):**

  - prod, month

- **F-VECT([F]):**

  - x_sum_quant, y_sum_quant

- **SELECT CONDITION-VECT([C]):**

  - x.prod=prod and x.month=month and x.year=2005

  - y.prod=prod

- **HAVING_CONDITION(G):**

  - y_sum_quant<>0

Technology Description

- **Programming language:**
    - Python 3.7
- **Compiler:**
    - Pyinstaller (Python Package)
- **DBMS:**
    - PostgreSQL 9.3.6
- **Packages:**
    - psycopg2, re, collections(defaultdict), itertools, argparse, sys
- **Operating System:**
    - Windows 10

Limitations

- Only EMF Query with the formatting structure

- No support for later grouping variable calculation based on previous one

- No support for MAX and MIN aggregation functions yet

- No support for minimal scanning

- No error checking for presence of tables

Sample 7_2

- Support +-*/ for [S] and [G]

- Support ">=" "<=" etc, logic comparison

- Scan n + 1 times of records

- Detect "avg" and add "sum and count" for MF_Vector

- **SELECT ATTRIBUTE(S):**

  - cust,prod, x_count_prod, x_avg_quant+1, y_avg_quant/2 ,x_avg_quant/y_avg_quant

- **NUMBER OF GROUPING VARIABLES(n):**

  - x, y

- **GROUPING ATTRIBUTES(V):**

  - cust,prod

- **F-VECT([F]):**

  - x_avg_quant, y_avg_quant, x_count_prod

- **SELECT CONDITION-VECT([C]):**

  - x.cust=cust and x.prod=prod

  - y.cust=cust and y.prod<>prod

- **HAVING_CONDITION(G):**

  - y_avg_quant<>0 and x_avg_quant>=0

Demo with python

Recap and Forward Looking

- Input EMF formatting txt file

- Collect schema and generate MF Structure

- Compute the outcome

- Build other aggregation function with optimal algorithm

- Further improve the syntax parsing part

stevens.edu

THANK YOU!