

**CZ 4031:**

**Database System Principles**

**Project 3:**

**Makeup Project**

Chen Hailin

**Principle**:

Query Execution Plan (QEP) is a concrete execution plan, generated based on SQL query by query optimizer in database. It has two steps: firstly, generate logical execution plan (involving a tree structure of relational algebra) and then choose concrete operation and algorithm to perform each step, making it into a QEP. In principle, there are factorial number of possible QEP tree for a given length of SQL query.

**Task**:

For two SQL queries with minor changes in WHERE/SELECT fields, find the change in two QEP trees. In fact, matching node correspondence in two QEP trees is not well-defined problem as various, completely different tree structures and nodes might be logically equivalent with a same SQL query. Thus, several assumptions and heuristics are made, based on QEP generator and targeted nodes.

Good news is we only deal with node change caused by WHERE/SELECT field change. I assume commercial QEP generator will perform “push selection” strategy so SELECT condition. Thus, I assume changes are reflected at leaf nodes for SELECT field and changes are reflected at scan nodes (‘Seq Scan’, ‘Index Scan’, etc) for WHERE field changes. Other changes include new node type (sort, aggregate) due to new algorithm and new join method (Hash/Merge/NL).

**Algorithm Details**:

1. Match Difference in SQL:

We parse SQL query and store the processed SQL query. To compare qep trees, we firstly parse the conditions for SELECT and WHERE. Then we get the newly added fields and deleted old fields.

1. Match SELECT/WHERE conditions in QEP trees

From the newly added and deleted fields located in (1), I match leaf nodes with SELECT fields and scan nodes with WHERE fields

1. Match JOIN/SCAN type change

We compare all JOIN/SCAN type of nodes in both qep trees and mark those with newly added and deleted types

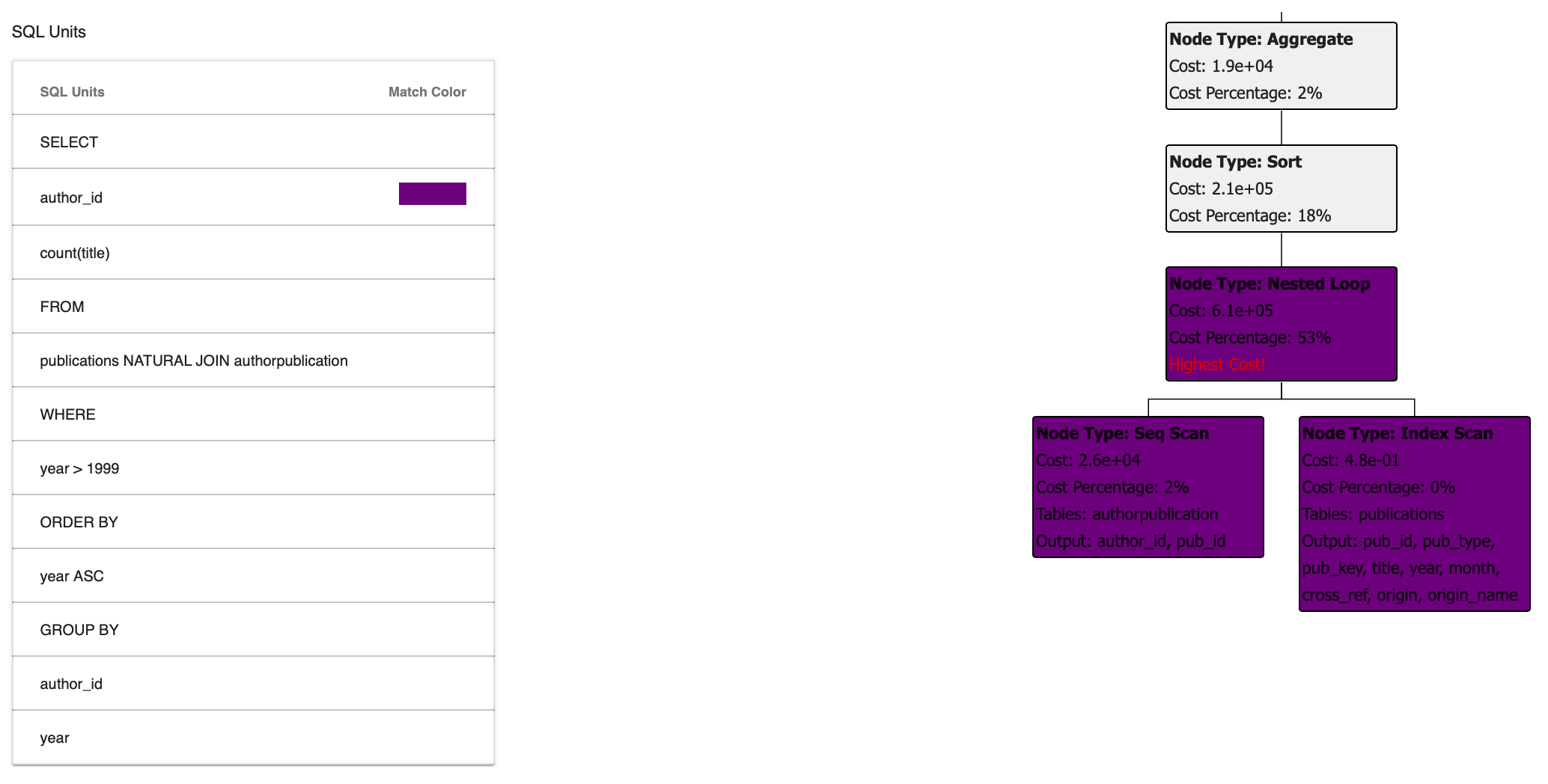
1. Match other type change

For any node type that is newly added/deleted, we will highlight them.

1. Visualization:

For all qep node with newly added SELECT/WHERE fields or newly added type, I will highlight them with GREEN color. For those deleted, I will highlight them with PURPLE color. All qep nodes highlighted by SELECT/WHERE directly will be linked with one cause SQL string (SELECT/WHERE condition). I will highlight the SQL string and its direct caused qep node to YELLOW color, when hovering on corresponding SQL string.

**Examples**



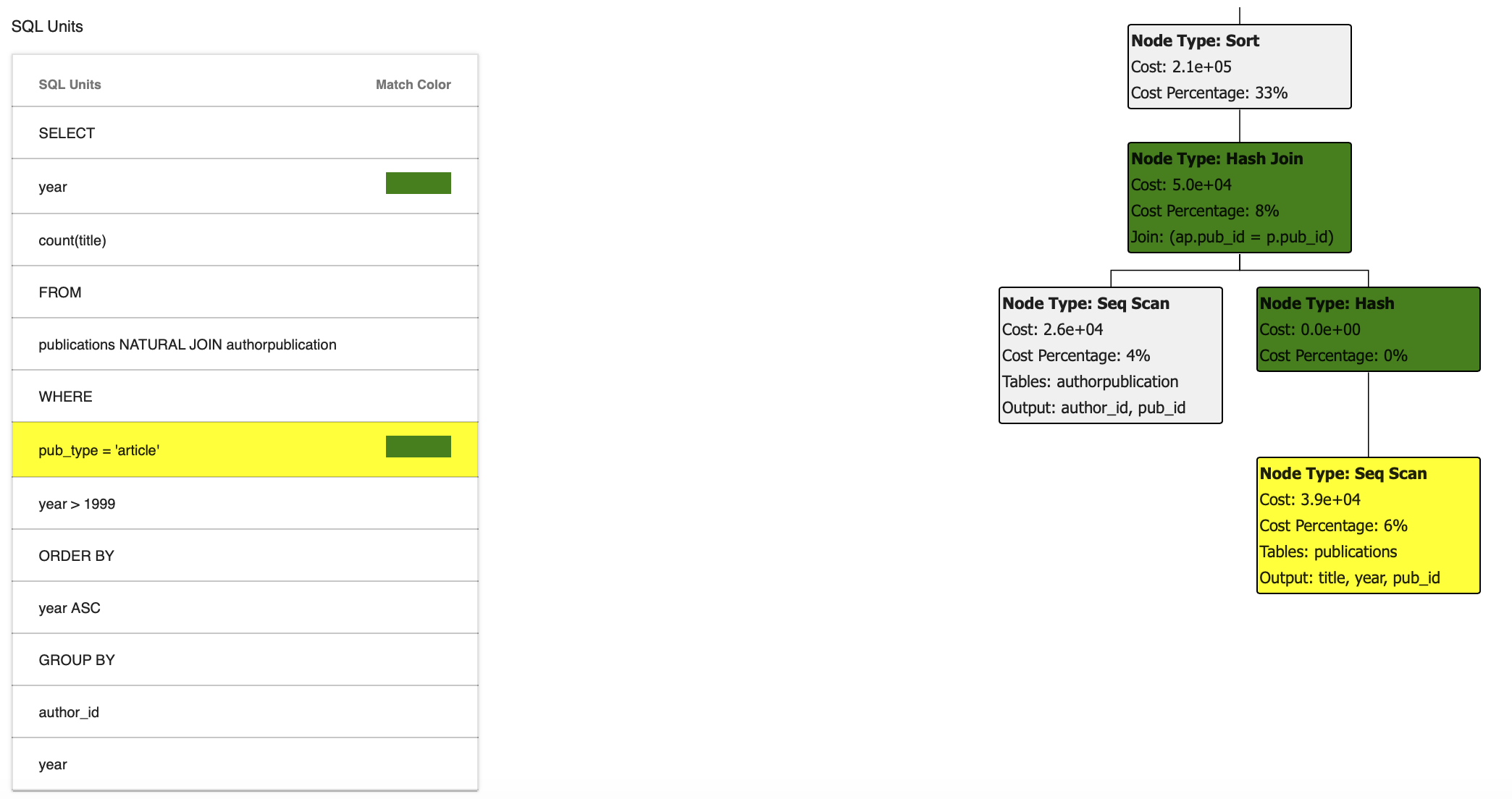
test\_B1:

old\_sql: SELECT count(title) as num\_articles, author\_id FROM publications NATURAL JOIN authorpublication where year>1999 GROUP BY author\_id, year order by year;

new\_sql: SELECT count(title) as num\_articles, year FROM publications NATURAL JOIN authorpublication where year>1999 and pub\_type='article' GROUP BY author\_id, year order by year;

The above showed old sql and old qep for test\_B1. Note the purpled blocks are deleted nodes due to sql change.

Below shows new sql and new qep for test\_B1. Note that greened blocks are newly added ones. The yellow blocks are caused by my mouse(not in screenshot) hovering on sql unit, indicating newly added pub\_type = ‘article’ caused to the corresponding Seq Scan node to select one more field as output.

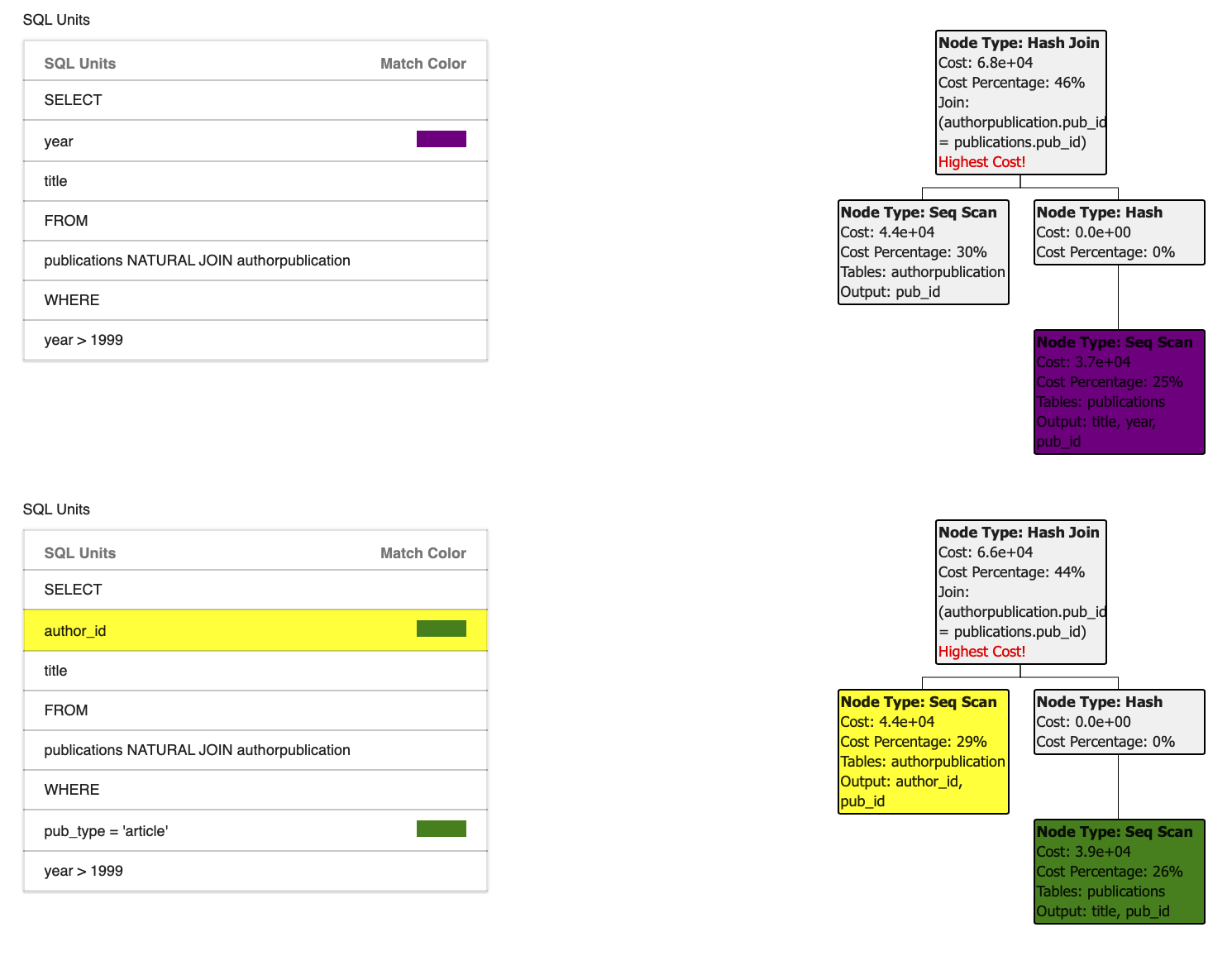


Below is another example.

Test\_example:

Old\_sql: SELECT title, year FROM publications NATURAL JOIN authorpublication where year>1999;

New\_sql: SELECT title, author\_id FROM publications NATURAL JOIN authorpublication where year>1999 and pub\_type='article';



The database I use for testing is the DBLP database used in project 1&2. However, since the algorithm is general for any case (as long as JSON format of qep input), it should be able to deal with more examples and more cases.

**Code Guide**

More Detailed version can be found in software\_installation\_guide.txt

In makeup\_test.py, you can find file paths for old/new sql and old/new qep JSON. I have included 4 test examples. Test\_example[0-2] have same old/new sql and old qep. But they have different new qep tree due to setting change in postgresql. Test\_exampleB represents one other example. You can add test example of similar format and change its root in makeup\_test.py. To run the project, use python 3.6 to run makeup\_test.py, you will then see webpage like I showed above at localhost:5000

For your information, the main code for makeup project lies app/sql\_parser.py(437p-636p) and qep\_parser.py(181p-282p), and app/template/compare\_visualize.html, which all are added after project 2.