CS3223 Database Systems Implementation
AY21/22 Semester 2
Experimental Write-Up
Team 37

Prof Tan Kian-Lee

Team Members	Student No.	Email
Kelvin Wong	A0201706U	e0415515@u.nus.edu
Moon Geonsik	A0210908L	e0484312@u.nus.edu
Wincent Tjoi	A0201480W	e0412905@u.nus.edu

CS3223 SimpleDB+ Experimental Writeup

We conducted the experiments with the following tables described in Figure 1, each containing about 50 tuples.

Tables	Attributes	Note:
Student	Sid, Sname, **MajorId**, GradYear	- **Bolded** attributes have btree index
Enroll	Eid, StudentId, Grade	- underscored attributes have hash index
Section	SectId, CourseId, Prof, **YearOffered**	
Course	Cid, Title, DeptId	
Dept	Did, Dname	

Figure 1: Table attributes with indexes shown

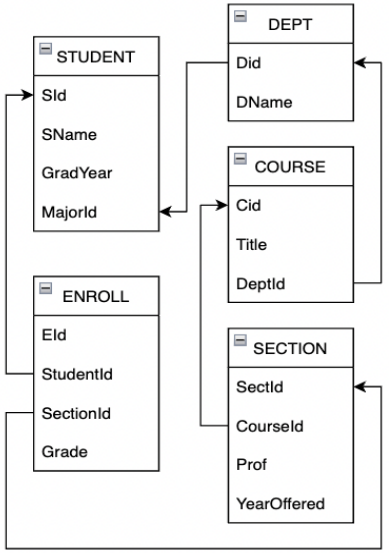


Figure 2: Schema for reference

Experiment 1 : 2-table join query

Query: ***select dname, title from dept, course where deptid < 450 and deptid = did***

Join Plan	Index-join	Merge-join	Hash-join	Nested-join
Run 1	0.25	9	11	0.31
Run 2	0.21	5	14	0.25
Run 3	0.18	4	9	0.21
Run 4	0.32	6	12	0.39
Run 5	0.28	5	11	0.35
Average Time (ms)	**0.25**	**5.8**	**11.4**	**0.30**

Figure 3: Timings for 2-table join query

Join Plan	Index-join	Merge-join	Hash-join	Nested-join
Blocks accessed	**11**	**18 estimated / 20 actual**	**36**	**17**
Analysis	When unrestricted to particular join algorithms, our SimpleDB+ selects the join plan with the lowest estimated number of blocks accessed. The timings in Figure 1 were relatively congruent with the number of blocks accessed by the joins. In an unrestricted execution plan, this results in index-join being chosen consistently when available, which gives merit to using the number of blocks accessed as a heuristic. The blocks accessed heuristic will successfully choose merge-join over hash-join, but if Heuristic 7 from the textbook was used and index-join was unavailable, the slower hash-join would have been used over merge-join. In terms of block access, hash-join has overhead from building hash tables, while merge-join has overhead from sorting and materializing the base tables. However, one flaw is that the estimation of mergejoin's blocks accessed can be inaccurate because the number of buffers available can change between the time of creating a plan, and time of actually executing the plan.			

	Nested-join actually performs on par with index-join here, but this is likely due to the relatively small size of the tables used which allows for disk caching to come into effect, reducing the number of blocks accessed by nested-join. With larger table sizes, nested-join will perform poorer.

Screenshots: 2-table index join, 2-table merge join, 2-table hash join, 2-table nested join

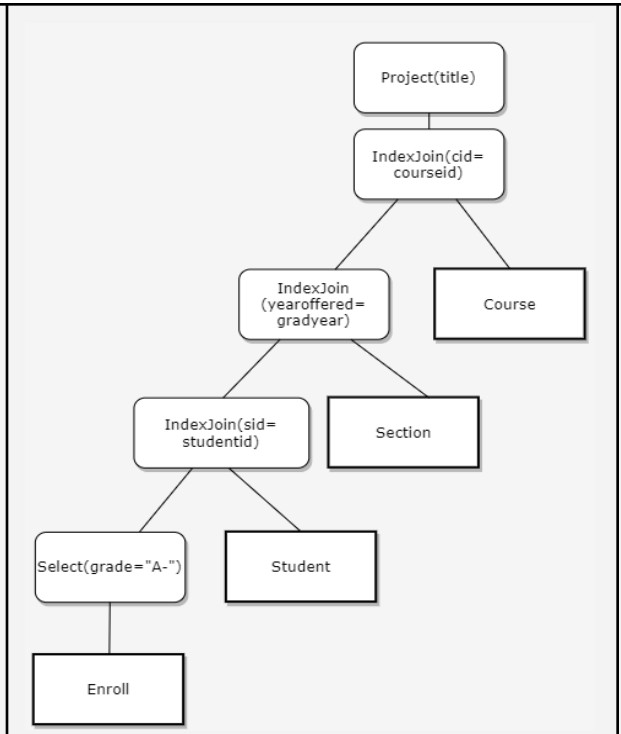
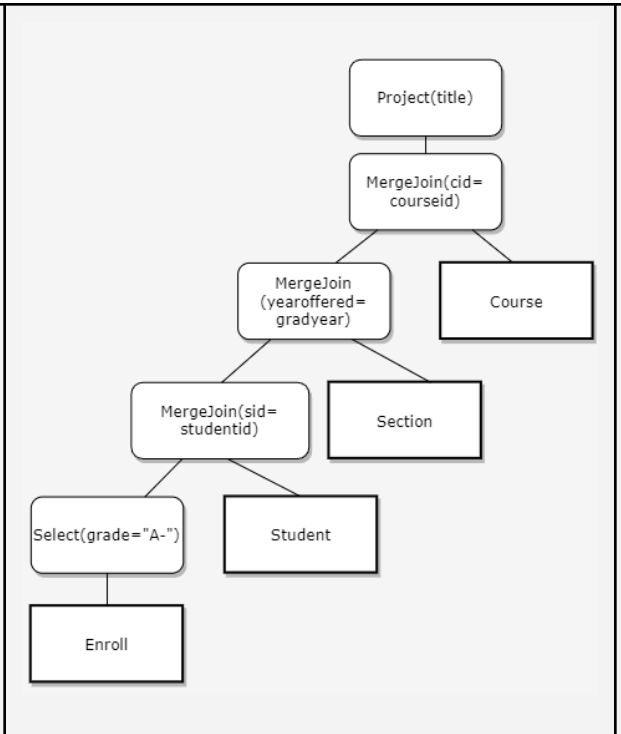
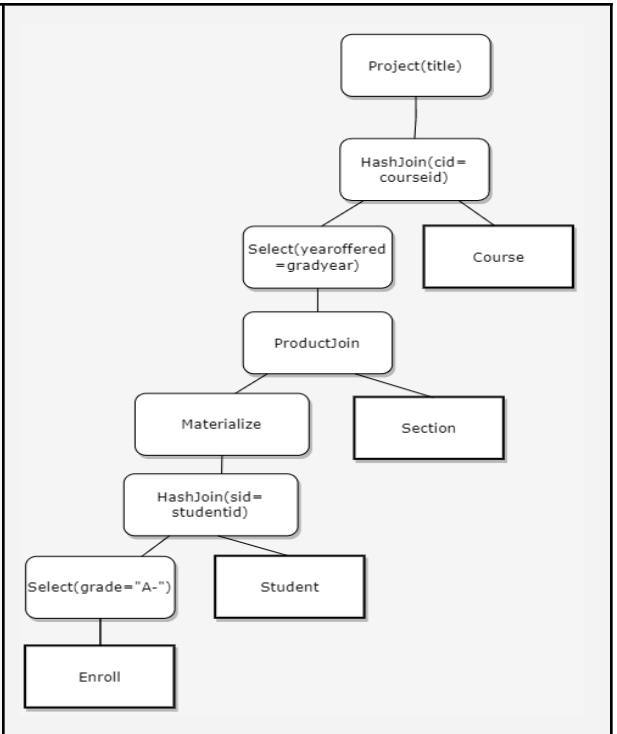
Experiment 2 : 4-table join query:

Query: ***select title from enroll, student, section, course where grade = 'A-' and studentid = sid and courseid = cid and yearOffered = gradyear***

Join Plan	Unrestricted	Merge-join	Hash-join
Run 1	0.78	157	55
Run 2	0.74	105	33
Run 3	0.68	103	41
Run 4	0.7	111	37
Run 5	0.82	132	43
Average Time (ms)	**0.62**	**121.6**	**41.8**

Figure 4: Timings for 4-table join query

Join Plan	Unrestricted	Merge-join	Hash-join
Blocks accessed	**35**	**46 estimated / 70 actual**	**42**

Plan Diagram			
Analysis	Based on the results, the blocks accessed by the plans are indicative of the time to execute them, similar to Experiment 1. On the other hand, merge-join only has a more drastic increase in query time and blocks accessed compared to the other 2. This is attributed to the exponential materialization and sorting overhead from the larger sized intermediate results compared to Experiment 1. Notably, the hash-join only execution plan has an additional materialization plan before the productjoin, which happens because of insufficient buffer. However, it is outweighed by the overhead from multiple materializations in merge-join only plan, causing it to cost more block accesses and execute slower than hash join plan. All 3 execution plans have the same join order of tables. This is because the heuristics used for join order is records output which is independent of the join algorithm.		

Screenshots: 4-table index join, 4-table merge join, 4-table hash join

Appendix

2-table index join

```
>>Indexjoin blocks accessed = 11
Table Plan used on table course
Select Plan used on predicate deptid<450
Table Plan used on table dept
Index join Plan used on did=deptid
Project Plan used on fields of dname, title
The total time taken for query is: 258333ns
```

2-table merge join

```
>>Mergejoin blocks accessed = 18
Table Plan used on table course
Select Plan used on predicate deptid<450
Sort Plan used
Table Plan used on table dept
Sort Plan used
Merge join plan used on deptid=did
Project Plan used on fields of dname, title
The total time taken for query is: 4ms
```

2-table hash join

```
>>Hashjoin blocks accessed = 36
Table Plan used on table course
Select Plan used on predicate deptid<450
Table Plan used on table dept
Hash join plan used on deptid=did (7 partitions created)
Project Plan used on fields of dname, title
The total time taken for query is: 10ms
```

2-table nested join

```
>>Nestedjoin blocks accessed = 17
Table Plan used on table course
Select Plan used on predicate deptid<450
Table Plan used on table dept
Nested join plan used on deptid=did
Project Plan used on fields of dname, title
The total time taken for query is: 390959ns
```

4-table index join

```
>>Indexjoin blocks accessed = 7
>>Indexjoin blocks accessed = 19
>>Indexjoin blocks accessed = 35
Table Plan used on table enroll
Select Plan used on predicate grade=A-
Table Plan used on table student
Index join Plan used on sid=studentid
Table Plan used on table section
Index join Plan used on yearoffered=gradyear
Table Plan used on table course
Index join Plan used on cid=courseid
Project Plan used on fields of title
The total time taken for query is: 746250ns
```

4-table merge join

```
>>Mergejoin blocks accessed = 24
>>Mergejoin blocks accessed = 26
>>Mergejoin blocks accessed = 46
Table Plan used on table enroll
Select Plan used on predicate grade=A-
Sort Plan used
Table Plan used on table student
Sort Plan used
Merge join plan used on studentid=sid
Sort Plan used
Table Plan used on table section
Sort Plan used
Merge join plan used on yearoffered=gradyear
Sort Plan used
Table Plan used on table course
Sort Plan used
Merge join plan used on courseid=cid
Project Plan used on fields of title
The total time taken for query is: 105ms
```

4-table hash join

```
>>Hashjoin blocks accessed = 30
Hashjoin: Not enough buffer size available for gradyear=yearoffered
Hashjoin failed: using productjoin instead
>>Productjoin blocks accessed = 7
>>Hashjoin blocks accessed = 42
Table Plan used on table enroll
Select Plan used on predicate grade=A-
Table Plan used on table student
Hash join plan used on studentid=sid (7 partitions created)
Materialized Plan created with fields [eid, studentid, sectionid, grade, sid, sname, majorid, gradyear]
Table Plan used on table section
Multibuffer Product Plan used between simpledb.materialize.MaterializePlan@6e8cf4c6 and simpledb.plan.TablePlan@12edcd21
Select Plan used on predicate yearoffered=gradyear
Table Plan used on table course
Hash join plan used on courseid=cid (7 partitions created)
Project Plan used on fields of title
The total time taken for query is: 33ms
```