David Schulz
CS 3860
Lab 6: Indexing
10/28/2020

Lab 6: Indexing

- INNODB
    - Loading gene_info50000.csv - 50000 records loaded
    - Loading gene2pubmed - 12917351 records loaded
    - Q1
        - First query – 5.37 sec
            - Keys: None
            - Rows: All from both tables
        - Creating primary key for gene_info – 0.4 sec
        - Creating primary key for gene2pubmed – 26.68 sec
        - Second query – 3.91 sec
            - Keys
                - Gene_info: Uses GeneID as the primary key
                - Gene2pubmed: None
            - Rows
                - Gene_info: 1
                - Gene2pubmed: All
            - Even though the gene2pubmed has primary keys, the query doesn't seem to use them. This could be because the primary key was set up with the column order reversed from the order they are in the table.
        - Removing primary key for gene2pubmed – 35.44 sec
        - Recreating primary key for gene2pubmed with reversed order – 25.94 sec
        - Third query – 0.04 sec
            - Keys
                - Gene_info: Uses GeneID as primary key
                - Gene2pubmed: Uses GeneID and PMID as primary keys
            - Rows
                - Gene_info: 1
                - Gene2pubmed: 6
        - Observations
            - The second query was faster than the first by about 2 seconds because the gene_info key was being used, but not the gene2pubmed key. The last query took almost no time because both keys were being used. The larger the table is when a primary key is added to it, the bigger of a time difference it makes.

    - Q2
        - First query – 0.18 sec
            - Keys: None
            - Rows: All
        - Creating index – 0.28 sec
        - Second query – 0.002 sec
            - Keys: Gene_info uses LocusTag as a key called gene_info_locustag
            - Rows: 1
        - Observations
            - The second query was 0.178 seconds faster because a key was being used to search the LocusTag column.
    - Q3
        - Range query – 0.006 sec
            - Keys: Uses the gene_info table's primary key
            - Rows: 526
        - Observations
            - Since the range being searched is GeneID, the table's primary key, it's very efficient to find the specific rows that fall within the range.
    - Q4
        - Inserting – 0.043 sec
            - Keys: None
            - Rows: None
        - Observations
            - No keys need to be used and no rows need to be accessed because the query is only inserting data.
    - Q5
        - Updating – 0.071 sec
            - Keys: Uses gene_info's gene_info_locustag key
            - Rows: 379
        - Observations
            - It's efficient to find all 379 rows where LocusTag is '-' because the gene_info_locustag key is being used.
- MyISAM
    - Loading gene_info50000.csv - 50000 records loaded
    - Loading gene2pubmed - 12917351 records loaded
    - Q1
        - First query – 22.26 sec
            - Keys: None
            - Rows: All from both tables

  - Creating primary key for gene_info – 0.51 sec
  - Creating primary key for gene2pubmed – 98.14 sec
  - Second query – 20.57 sec
    - Keys
      - Gene_info: Uses GeneID as primary key
      - Gene2pubmed: None
    - Rows
      - Gene_info: 1
      - Gene2pubmed: All
  - Removing primary key for gene2pubmed – 59.53 sec
  - Recreating primary key for gene2pubmed with reversed order – 101.15 sec
  - Third query – 0.0006 sec
    - Keys
      - Gene_info: Uses GeneID as primary key
      - Gene2pubmed: Uses GeneID and PMID as primary keys
    - Rows
      - Gene_info: 1
      - Gene2pubmed: 5
  - Observations
    - The second query was faster than the first by about 2 seconds because the gene_info key was being used, but not the gene2pubmed key. The last query took almost no time because all keys for both tables were being used.
- Q2
  - First query – 0.18 sec
    - Keys: None
    - Rows: All
  - Creating index – 0.63 sec
  - Second query – 0.0007 sec
    - Keys: Gene_info uses LocusTag as a key called gene_info_locustag
    - Rows: 1
  - Observations
    - The second query was 0.6203 seconds faster because a key was being used to search the LocusTag column.
- Q3
  - Range query – 0.006 sec
    - Keys: Uses the gene_info's primary key
    - Rows: 695
  - Observations

- Since the range being searched is GeneID, the table’s primary key, it’s very efficient to find the specific rows that fall within the range. The number of rows that were searched was larger than the actual number of rows that fell within the range this time.

- Q4
  - Inserting – 0.012 sec
    - Keys: None
    - Rows: None
  - Observations
    - No keys need to be used and no rows need to be accessed because the query is only inserting data.
- Q5
  - Updating – 0.032 sec
    - Keys: Uses gene_info’s gene_info_locustag key
    - Rows: 135
  - Observations
    - It’s efficient to find all 379 rows where LocusTag is ‘-‘ because the gene_info_locustag key is being used. However, only 135 rows needed to be accessed this time.

- MEMORY
  - Loading gene_info50000.csv - 50000 records loaded
  - Loading gene2pubmed - 12917351 records loaded
  - Q1
    - First query – 0.44 sec
      - Keys: None
      - Rows: All
    - Creating primary key for gene_info – 1.1 sec
    - Creating primary key for gene2pubmed – 7.4 sec
    - Second query – 0.29 sec
      - Keys
        - Gene_info: Uses GeneID as primary key
        - Gene2pubmed: None
      - Rows
        - Gene_info: 1
        - Gene2pubmed: All
    - Removing primary key for gene2pubmed – 1.37 sec
    - Recreating primary key for gene2pubmed with reversed order – 7.63 sec
    - Third query – 0.3 sec
      - Keys
        - Gene_info: Uses GeneID as primary key

- Gene2pubmed: None
- Rows
  - Gene_info: 1
  - Gene2pubmed: All
- Observations
  - The second query was faster than the first by about 0.15 seconds because the gene_info key was being used, but not the gene2pubmed key. The second and third queries were the same because the way they accessed the rows didn't change. Changing the order of the columns to create the primary key doesn't affect it when the primary key is a hash.

- Q2
  - First query – 0.24 sec
    - Keys: None
    - Rows: All
  - Creating index – 1.24 sec
  - Second query – 0.0008 sec
    - Keys: Gene_info uses LocusTag as a key called gene_info_locustag
    - Rows: 2
  - Observations
    - The second query was 0.2302 seconds faster because a key was being used to search the LocusTag column.
- Q3
  - Range query – 0.24 sec
    - Keys: None
    - Rows: All
  - Observations
    - Since the primary key isn't being used this time, it takes a little longer.
- Q4
  - Inserting – 0.038 sec
    - Keys: None
    - Rows: None
  - Observations
    - No keys need to be used and no rows need to be accessed because the query is only inserting data.
- Q5
  - Updating – 0.025 sec
    - Keys: Uses gene_info's gene_info_locustag key
    - Rows: 2

        - Observations
            - It’s efficient to find all 379 rows where LocusTag is ‘-‘ because the gene_info_locustag key is being used. However, only 2 rows needed to be accessed this time.

Question #	INNODB Time (sec)	MyISAM Time (sec)	MEMORY Time (sec)
1	0.04	0.0006	0.3
2	0.002	0.0007	0.0008
3	0.006	0.006	0.24
4	0.043	0.012	0.038
5	0.071	0.032	0.025

- Observations
    - The MyISAM engine was generally faster than the INNODB engine. For the MEMORY engine, some queries were as fast as the other two, but the ones that were slower than the other two were slower by a large difference. This is because of the type of primary key MEMORY used, hash, instead of a tree. The difference between the types of primary keys for INNODB and MyISAM was much less significant than the difference with the type of primary key for MEMORY.