UNIVERSITY COMPUTER SCIENCE • EXERCISE SHEET

8.4 Indexes and query plans

Name: _____ Class: _____ Date: _____ **Total: 24 marks**

KEY WORDS B-tree B • query plan

Objective

Price lookups by fan-out and read the plan the engine prints.

1 Key knowledge and worked examples

■ The arithmetic

Fan-out m, height h covers m^h keys: fan-out 100, height 3 covers a million – 3 node reads per lookup ✓. Order-4 tree, 15 keys: height 2 ✓.

■ Worked example: the plan

`EXPLAIN QUERY PLAN SELECT title FROM Course WHERE credits = 20:` `SCAN Course` unindexed; `SEARCH Course USING INDEX idx (credits=?)` with the index ✓.

2 Practice

2.1 Fan-out 200, height 3: how many keys? [3]

2.2 An order-8 tree over 500 keys: give the height. [3]

2.3 A table with 4 indexes: how many structure writes per insert? [3]

3 Exam-style questions

3.1 A table of 2×10^7 rows uses an order-128 B-tree on its key.

Compute the height needed – the smallest h with 128^h at least 2×10^7 – and the reads per lookup. Compute the reads a full scan takes. Compute the write amplification of maintaining 4 indexes on this table per insert. State what `EXPLAIN QUERY PLAN` would print for a key-equality query with and without the index. [8]

3.2 Two queries run on the course schema: `WHERE UPPER(title) = 'DATABASES'` and `WHERE title = 'Databases'`, with a plain index on title.

State which query the index serves and why the other defeats it. Give the general rule about functions over indexed columns. Name the index form that would serve the first query, and its cost relative to a plain index. [7]

Solutions

2.1

- $200^3 = 8 \times 10^6$ keys ✓.

2.2

- $8^3 = 512 \geq 500$: height 3 ✓.

2.3

- Table + 4 trees = 5 structure writes ✓.

3.1

- $128^3 \approx 2.1 \times 10^6 < 2 \times 10^7$ is not enough; $128^4 \approx 2.7 \times 10^8$ covers it – height 4 ✓.
- Reads per lookup: 4 node reads ✓.
- Full scan: 2×10^7 row reads ✓.
- Write amplification: $1 + 4 = 5$ structures per insert ✓.
- Plan without index: `SCAN table`; with the key index: `SEARCH … USING INDEX … (key=?)` ✓.

3.2

- The plain equality `title = 'Databases'` uses the index ✓.
- `UPPER(title) = …` computes over the column: the stored sorted keys are not the values being compared – the engine scans ✓.
- Rule: a function over an indexed column defeats a plain index ✓.
- An expression index on `UPPER(title)` serves it – one more tree to maintain on every write, the same loan at higher interest ✓.