Exercise walk-through

8.4
Indexes and query plans

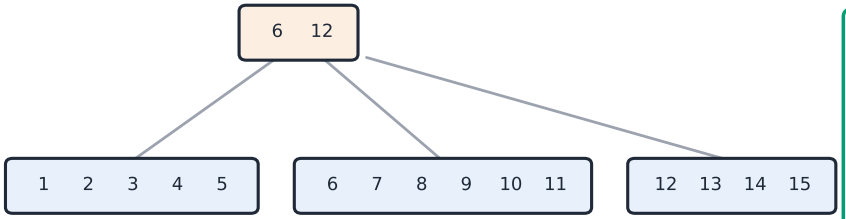
University Computer Science

Indexes and query plans: the plan tells the truth

1. **Tree** Order-4, 15 keys, 2 levels: any key in 2 reads; fan-out 100 covers a million keys at height 3.
2. **Plan** EXPLAIN **QUERY PLAN**: without an index, SCAN Course; with INDEX ON Course(credits), SEARCH USING INDEX - 6 rows instant, 6 million rows 4 reads versus a scan.
3. **Tax** Table + 3 trees = 4 structure writes per insert; a 5-index log at 10 million inserts daily maintains 50 million entries.

B-tree and query plan

two levels, fifteen keys



order-4 B-tree: any key in 2 reads

fan-out: 100^3 = a million keys in 3 reads

the plan tells the truth

EXPLAIN QUERY PLAN

no index:

SCAN Course

with INDEX ON Course(credits):

SEARCH USING INDEX idx (credits=?)

6 rows: both instant; 6 million: 4 reads

every index is a loan the writes repay: table + 3 trees = 4 structures per insert

Left: the **B-tree**, two levels for 15 keys. Right: the plan that uses it.

Exercises 2.1·2.3

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]

Exercises 2.1·2.3

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

2.1

- $200^3 = 8 \times 10^6$ keys ✓.

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

2.2

- $8^3 = 512 \geq 500$: height 3 ✓.

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

2.3

- Table + 4 trees = 5 structure writes ✓.

Exercise 3.1

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]

Exercise 3.1

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]

3.1

- $128^3 \approx 2.1 \times 10^6 < 2 \times 10^7$ 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=?)` ✓.

Exercise 3.2

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]

Exercise 3.2

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]

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 ✓.