CSCI317 Database Performance Tuning

Indexing Examples

Dr Janusz R. Getta

School of Computing and Information Technology -
University of Wollongong

Indexing Examples

Outline

Examples

Examples

Consider a relational table
`PRODUCT(PNUM, PNAME, PRICE, MANUFACTURER)`

Assume that

- the table contains 10^5 rows,
- an attribute `PNUM` is a primary key,
- the products are manufactured by `100` manufacturers,
- the table contains `1000` product names,
- an average number of rows per disk block (blocking factor) is `50`
- a database administrator created a non-clustered B*-tree index on attribute `MANUFACTURER`,
- a fanout of B*-tree is equal to `f`
- a width (total number of blocks at leaf level) of B*-tree implementing an index on the primary key is equal to w_p
- a width (total number of blocks at leaf level) of B*-tree implementing an index on an attribute `MANUFACTURER` is equal to w_m

Example 1

To compute a query

Projection query, no index can be used

```
SELECT PNAME, PRICE
FROM   PRODUCT;
```

we have to read $10^5/50$ (total number of rows/blocking factor) blocks because the entire relational table PRODUCT must be read

A fragment of respective query processing plan

A fragment of query processing plan

```
-------------------------------------------- ...
| Id | Operation          | Name    |
-------------------------------------------- ...
| 0  | SELECT STATEMENT   |         |
| 1  |  TABLE ACCESS FULL | PRODUCT |
-------------------------------------------- ...
```

Example 2

To compute a query

Selection and projection query, index on primary key can be used

```
SELECT PNAME, PRICE
FROM   PRODUCT
WHERE PNUM = 12345;
```

we have to read $[\log_f 10^5 +1] + 1$ blocks because the primary key `PNUM` is automatically indexed

A fragment of respective query processing plan

A fragment of query processing plan

```
-----------------------------------------------------  ...
| Id | Operation                    | Name          |
-----------------------------------------------------  ...
| 0  | SELECT STATEMENT             |               |
| 1  |  TABLE ACCESS BY INDEX ROWID | PRODUCT       |
|*2  |   INDEX UNIQUE SCAN          | PRODUCT_PKEY |
-----------------------------------------------------  ...
```

Example 3

To compute a query

Selection and projection query, index on primary key can be used

```
SELECT PNAME, PRICE
FROM   PRODUCT
WHERE PNUM = 1234567 AND
      PRICE <= 20.0;
```

we have to read $[\log_f 10^5 +1] + 1$ blocks because the primary key `PNUM` is automatically indexed

A fragment of respective query processing plan

A fragment of query processing plan

```
----------------------------------------------------- ...
| Id | Operation                    | Name          |
----------------------------------------------------- ...
| 0  | SELECT STATEMENT             |               |
| 1  |  TABLE ACCESS BY INDEX ROWID | PRODUCT       |
|*2  |   INDEX UNIQUE SCAN          | PRODUCT_PKEY  |
----------------------------------------------------- ...
```

Example 4

To compute a query

Selection and projection query, index on primary key cannot be used due to disjunction (OR)

```
SELECT PNAME, PRICE
FROM   PRODUCT
WHERE  PNUM = 1234567 OR
       PRICE <= 20.0;
```

we have to read 10^5/50 (total number of rows/blocking factor) blocks because the entire relational table PRODUCT must be read

A fragment of query processing plan

```
-------------------------------------------- ...
| Id | Operation           | Name     |
-------------------------------------------- ...
| 0  | SELECT STATEMENT    |          |
| 1  |  TABLE ACCESS FULL  | PRODUCT  |
-------------------------------------------- ...
```

Example 5

To compute a query

Selection and projection query, index on MANUFACTURER can be used

```
SELECT PNAME, PRICE
FROM   PRODUCT
WHERE MANUFACTURER = 'GoldenBolts';
```

we have to read
$[\log_f 10^2 + 1] + ((10^5/100) + 10^5/(100*50))/2$ data blocks

Index processing: $[\log_f 10^2 + 1]$ read block operations

Total number of row identifiers found: $10^5/100$

Table processing:

- The best case (full clustering: $10^5/(100*50)$ read block operations
- The worst case (each row in a different block): $10^5 /100$ read block operations

Example 5 (continuation)

To compute a query

Selection and projection query, index on MANUFACTURER can be used

```
SELECT PNAME, PRICE
FROM   PRODUCT
WHERE MANUFACTURER = 'GoldenBolts';
```

we have to read

$[\log_f 10^2 + 1] + ((10^5/100) + 10^5/(100*50))/2$ data blocks

A fragment of query processing plan

```
---------------------------------------------------- ...
| Id | Operation                    | Name    |
---------------------------------------------------- ...
| 0  | SELECT STATEMENT             |         |
| 1  |  TABLE ACCESS BY INDEX ROWID | PRODUCT |
|*2  |   INDEX RANGE SCAN           | PIDX    |
---------------------------------------------------- ...
```

Example 6

To compute a query

Counting the rows, and index on primry key can be used

```
SELECT COUNT(*)
FROM PRODUCT;
```

we have to read w_p data blocks (width of leaf level on an index on primary key)

Index processing: w_p read block operations

Counting row identifiers: 0 read block operations, counting is performed in transient memory

A fragment of query processing plan

```
----------------------------------------------- ...
| Id | Operation             | Name         |
----------------------------------------------- ...
| 0  | SELECT STATEMENT      |              |
| 1  |  SORT AGGREGATE       |              |
| 2  |   INDEX FAST FULL SCAN| PRODUCT_PKEY |
----------------------------------------------- ...
```

Example 7

To compute a query

Counting the rows with selection, an index on MANUFACTURER can be used

```
SELECT COUNT(*)
FROM  PRODUCT;
WHERE MANUFACTURER = 'GoldenBolts';
```

we have to read $[\log_f 10^2 +1]$ data blocks

Index processing: $[\log_f 10^2 +1]$ read block operations

Counting row identifiers: 0 read block operations, counting is performed in transient memory

A fragment of query processing plan

```
--------------------------------------------- ...
| Id | Operation          | Name   |
--------------------------------------------- ...
| 0  | SELECT STATEMENT   |        |
| 1  |  SORT AGGREGATE    |        |
|*2  |   INDEX RANGE SCAN | PIDX   |
--------------------------------------------- ...
```

Example 8

To compute a query

```
                                          Projection and selection with patter matching

SELECT MANUFACTURER
FROM   PRODUCT
WHERE MANUFACTURER LIKE 'M%';
```

we have to read w_m data blocks (width of a leaf level of an index on MANUFACTURER)

Index processing: w_m read block operations

```
                                                    A fragment of query processing plan

-------------------------------------------- ...
| Id | Operation         | Name |
-------------------------------------------- ...
| 0  | SELECT STATEMENT  |      |
|*1  |  INDEX RANGE SCAN | PIDX |
-------------------------------------------- ...
```

Example 9

To compute a query

Projection query with elimination of duplicates, an index on MANUFACTURER can be used

```
SELECT DISTINCT MANUFACTURER
FROM   PRODUCT;
```

we have to read w_m data blocks

Index processing: w_m read block operations (horizontal traversal through leaf level on an index on MANUFACTURER)

A fragment of query processing plan

```
---------------------------------------------- ...
| Id | Operation              | Name          |
---------------------------------------------- ...
| 0  | SELECT STATEMENT       |               |
| 1  |  SORT UNIQUE           |               |
| 2  |   INDEX FAST FULL SCAN| PRODUCT_PKEY  |
---------------------------------------------- ...
```

Example 10

To compute a query

Aggregation query, an index on MANUFACTURER can be used

```
SELECT MANUFACTURER, COUNT(*)
FROM PRODUCT
GROUP BY MANUFACTURER;
```

we have to read w_m data blocks

Index processing: w_m read block operations

A fragment of query processing plan

```
---------------------------------------------- ...
| Id | Operation            | Name          |
---------------------------------------------- ...
| 0  | SELECT STATEMENT     |               |
| 1  |  SORT GROUP BY NOSORT|               |
| 2  |   INDEX FULL SCAN    | PRODUCT_PKEY  |
---------------------------------------------- ...
```

Examples (continuation)

Consider a relational table
`PRODUCT(PNUM, PNAME, PRICE, MANUFACTURER)`

Assume that

- a database administrator created a non-clustered B*-tree index on attribute `PNAME`,
- the table contains `1000` distinct product name
- a fanout of B*-tree is equal to `f`
- a width (total number of blocks at leaf level) of B*-tree implementing an index on an attribute `PNAME` is equal to w_n

Example 10

To compute a query

```
  Selection query with conjunction WHERE condition, both indexes on MANUFACTURER and PNAME can be used
SELECT *
FROM PRODUCT
WHERE PNAME = 'bolt' AND
      MANUFACTURER = 'Golden Bolts';
```

we have to read $[\log_f 1000\ +1]\ +\ [\log_f 100\ +1]\ +\ 1$ data blocks

Index processing: $[\log_f 1000\ +1]$ read block operations to process an index on PNAME

The total number of row identifiers found: $10^5/1000$

Index processing: $[\log_f 100\ +1]$read block operations to process an index on MANUFACTURER

The total number of row identifiers found: $10^5/100$

Intersection of the sets of row identifiers obtained from index processing: 0 read block operations

Example 10

Total number of row identifiers found: 10^5/(1000*100) = 1

Table processing: 1 read block operations

Example 11

To compute a query

Selection query with disjunction WHERE condition, both indexes on MANUFACTURER and PNAME can be used

```
SELECT *
FROM PRODUCT
WHERE PNAME = 'bolt' OR
      MANUFACTURER = 'Golden Bolts';
```

we have to read $[\log_f 1000 +1] + [\log_f 100 +1] + ((10^2 + 10^3 - 1) + (10^2 + 10^3 - 1)/50)/2$ data blocks

Index processing: $[\log_f 1000 \ +1]$ read block operations to process an index on PNAME

The total number of row identifiers found: $10^5/1000$

Index processing: $[\log_f 100 \ +1]$read block operations to process an index on MANUFACTURER

The total number of row identifiers found: $10^5/100$

Example 11

Union of the sets of row identifiers obtained from index processing: 0 read block operations

Total number of row identifiers found: $(10^5/1000) + (10^5/100) - (10^5/(1000*100)) = 10^2 + 10^3 - 1$

Table processing: $((10^2 + 10^3 - 1) + (10^2 + 10^3 - 1)/50)/2$ read block operations

Example 12

To compute a query

Selection query with a range condition, an index on PNAME can be used

```
SELECT  *
FROM PRODUCT
WHERE PNAME > 'bolt';
```

we have to read $[\log_f 1000 + 1] + w_p * n / 10^5 + (n + n/50)/2$ data blocks

Vertical index processing: $[\log_f 1000 + 1]$ read block operations to process an index on `PNAME`

Horizontal index processing: $w_p * n / 10^5$ where w_p is the total number of leaf level blocks and n is the total number of rows that satisfy a condition `PNAME > 'bolt'`

References

Cookbook, How to measure and how to improve performance of database applications, how to choose the best index, how to analyze index structures ?

Ramakrishnan R., J. Gehrke Database Management Systems, chapters 8.1-8.3

Lightstone, S., Teorey T., Nadeau T., Physical Database Design, The Database Professional's Guide to Exploiting

Indexes, Views, Storage, and More, Morgan Kaufmann Publishers, 2007, chapter 4