Mentimeter

Part 1 recap quiz

Mentimeter

What is a relation?

Mentimeter

A relation can be seen as a table of values. Something like this:

Tutorials:

ID	*site*	*tutorial*	*topic*
1	w3schools	SQL_2003STD	Databases
2	w3schools	HTML_5	WebDev
3	w3schools	CSS_3	WebDev
4	w3resource	SQL_2003STD	Databases
5	w3resource	MySQL	Databases

- **Tutorials**: relation name
- *ID, site, tutorial, topic*: attribute names (or just attributes),
  collectively: signature
- Relation name + signature: relation schema (or just relation)
- Rows: tuples, records, or instances
  collectively: relation state
- Elements of tuples: attribute values (or just attributes as well)

Mentimeter

What are the core operations of the (set) relational algebra?

Mentimeter

- Domain: Relations (up to their names)
- Core operations:
  - (set) union $\cup$
  - (set) difference $\setminus$
  - selection σ_C
  - projection π_{Atts}
  - Cartesian product $\times$ (a.k.a. cross product, cross join)
  - renaming ϱ_N

Mentimeter

Which of the core relational algebra operations are unary, which are binary and which are parametrized?

Mentimeter

- Domain: Relations (up to their names)
- Core operations:
  - (set) union $\cup$
  - (set) difference $\setminus$
  - selection σ_C
  - projection π_{Atts}
  - Cartesian product $\times$ (a.k.a. cross product, cross join)
  - renaming ϱ_N

Mentimeter

Why do we need bag relational algebra?

Mentimeter

What is missing in our core RA to cover all SQL functionality?

1. Value invention (`SELECT R.A+5 FROM R`)
2. NULLs (in particular, for `OUTER JOIN`s):
3. Recursion (`WITH RECURSIVE`)
4. Bag relations and aggregates (`SELECT SUM(R.A) FROM R`)
5. Ordered relations (`ORDER BY`, `LIMIT`)

... ...

Mentimeter

What are the differences between generalised bag projection (bag relational algebra) and projection (relational algebra)?

Mentimeter

Main differences

- duplicates allowed (bags not sets)
- includes renaming
- includes value invention

Mentimeter

Generalised Projection: unary, one parameter (example)

Projection + renaming + value invention using expressions

$\pi_{site,\ topic \to area,\ ID+5 \to IDplus}$ (**Tutorials**:)

ID	*site*	*tutorial*	*topic*
1	w3schools	SQL_2003STD	Databases
2	w3schools	HTML_5	WebDev
3	w3schools	CSS_3	WebDev
4	w3resource	SQL_2003STD	Databases
5	w3resource	MySQL	Databases

$=$

$\pi_{site,\ topic \to area,\ ID+5 \to IDplus}$(**Tutorials**):

IDplus	*site*	*area*
6	w3schools	Databases
7	w3schools	WebDev
8	w3schools	WebDev
9	w3resource	Databases
10	w3resource	Databases

Mentimeter

Generalised Projection for bags (including renaming): unary, one parameter (example only)

Also as in set case (outputs the same number of tuples as in input):

$\pi^{b}_{site,\ topic \to area,\ ID+5 \to IDplus}$ (**Tutorials**:

ID	*site*	*tutorial*	*topic*
1	w3schools	SQL_2003STD	Databases
2	w3schools	HTML_5	WebDev
3	w3schools	CSS_3	WebDev
4	w3resource	SQL_2003STD	Databases
4	w3resource	SQL_2003STD	Databases

)

$=$

$\pi^{b}_{site,\ topic \to area,\ ID+5 \to IDplus}$**(Tutorials)**:

IDplus	*site*	*area*
6	w3schools	Databases
7	w3schools	WebDev
8	w3schools	WebDev
9	w3resource	Databases
9	w3resource	Databases

Mentimeter

What are the two bag-specific operations?

Mentimeter

Bag specific operation

Duplicate elimination

Grouping with aggregation

Mentimeter

What is the difference between a dense and a sparse index?

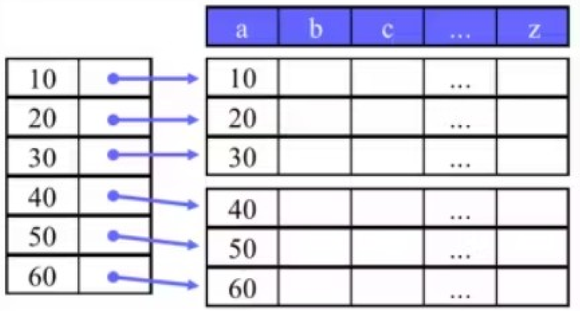
Mentimeter

Primary Dense Index

Primary Dense Index on the primary (or candidate) key attribute A in a relation R stored as sorted by A:

- sorted table with two "attributes":
  primary key value, pointer to the tuple with this value
- each tuple in R has a pair in the index

Example:



Benefit: index itself is also stored in secondary memory, but with much fewer blocks

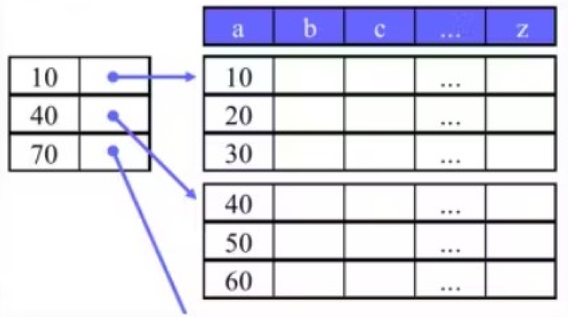
Mentimeter

Primary Sparse Index

Primary Sparse Index on the primary key attribute A in a relation R stored as sorted by A:

- sorted table with two "attributes":
  primary key value, pointer to the tuple with this value
- only the first tuple in each block of R has a pair in the index

Example:



Benefit: index itself is also stored in secondary memory, but with even fewer blocks

Mentimeter

What is the main difference between a primary and clustering index?

- The primary index is a dense index while the clustering index is sparse ×
- The clustering index works on several columns while the primary index works on only one column ×
- The clustering index allows for non-unique attribute values while the primary index is constrained to unique values ✓

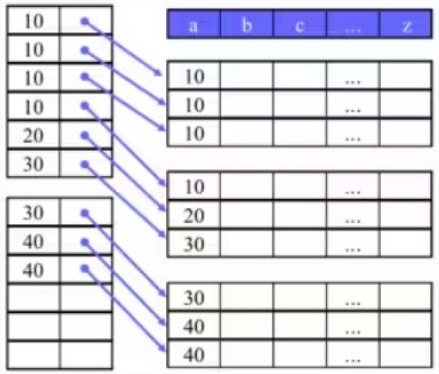
Mentimeter

Clustering Dense Index

Clustering Dense Index on an attribute A (any!) in a relation R stored as sorted by A:

- sorted table with two "attributes":
  attribute value, pointer to the tuple with this value
- each tuple in R has a pair in the index

Example:



Not much different from primary dense index

Mentimeter

Order the following operations of the query processor in the order they are applied on an input query: compilation, evaluation, optimization, parsing

Mentimeter

Correct

parsing, compilation, optimization, evaluation

Mentimeter

SQL (DQL) query processor in action

```
SELECT DISTINCT title FROM StarsIn WHERE starName IN
  (SELECT name FROM MovieStar WHERE birthDate LIKE '%1960');
```

⇣ parsing

(parse tree)

⇣ compiliation

$$\pi_{title}(\sigma_{starName=name}(\mathbf{StarIn} \times \pi_{name}(\sigma_{birthDate\ LIKE\ '\%1960'}(\mathbf{MovieStar}))))$$

⇣ optimisation

$$\pi_{title}(\mathbf{StarIn} \bowtie_{starName=name} \pi_{name}(\sigma_{birthDate\ LIKE\ '\%1960'}(\mathbf{MovieStar})))$$

⇣ evaluation over tables (next time)

(Result)

Mentimeter

How do we calculate the cost function for a logical query plan?

Mentimeter

Sum of estimates

"the sum of estimates of sizes (in bytes) of the intermediate results"

Mentimeter

What evaluation algorithm for the natural join has the weakest requirement?

Mentimeter

Nested-loop natural join

Natural join $R \bowtie S$

generalisation of one-pass

Apply one-pass join to the parts of the smaller of R and S separately, using $MMBuffers - 1$ buffers:

Assuming $Blocks(R) \leq Blocks(S)$,
Repeat:

1. load the next $MMBuffers - 1$ blocks of $Blocks(R)$
   into the $MMBuffers - 1$ buffers
2. apply one-pass join to these buffers and full S

Until all blocks of R are processed

Cost: $Blocks(R) + Blocks(S) * \lceil Blocks(R)/(MMBuffers - 1) \rceil$

Requirement: $2 \leq MMBuffers$

(essentially, no space requirement, but quadratic)

Mentimeter

What is the drawback of hash-based algorithms for query evaluation?

Mentimeter

Hash function

Good hash function is not always achievable

Mentimeter

What is the disadvantage of index based algorithms for query evaluation?

Mentimeter

Indexes

- might not have indexes on all necessary attributes
- too expensive to have indexes on all attributes