

1. Übung

Timo Bergerbusch 344408, Andreas Nikulin 332065

19.10.2017

1 Exercise 1.1

1. Layers:

Logical data structures:

concepts: translate and optimize queries

interface: set-oriented interface: relations, tuples, views

Logical access structures:

concepts: manage cursor, sort components and dictionary

interface: record oriented interface: records, sets, keys, access paths

Storage Structures:

concepts: manage record and index

interface: internal record interface: records, B* trees

Page assignment:

concepts: manage buffer and segments

interface: system buffer interface: pages, segments

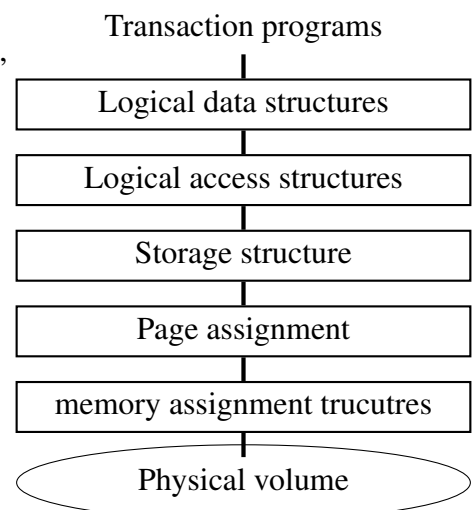
Memory assignment structures:

concepts: manage files and external memory

interface: file interface: blocks, files

physical volume:

interface: device interface: tracks, cylinders, channels



2. Order: $e \rightarrow b \rightarrow d \rightarrow a \rightarrow c$

3. (a) **data independence:** the view on the data is independent of its organized structure inside of the DB

Physical data independence: the underlying logical organization is independent of the physical representation. So restructuring or changing the implemented structure does not affect the schema

logical data independence: the logical schema might change without any affect on the external schema

(b) Data independence is important because it can provide an encapsulated split between development of programs on an external given structure independent of its internal handling.

(c) answer:

| Layer | What is hidden? | Problems: Due to high specialization, functionality of operating system often not usable <ul style="list-style-type: none"> • Segment-file mapping • Paging • Shadow memory • Buffer management • Dispatching |
|------------------------------|--|--|
| Logical data structures | Position indicator and explicit relations in the schema | |
| Logical access paths | Number and kind of the physical access paths; internal representation of records | |
| Storage structures | Management of buffers, logging | |
| Page assignment structures | File mapping, indirect page assignment | |
| Memory assignment structures | Technical features and technical details of external storage media | |

2 Exercise 1.2

1. relational algebra

- (a) $\pi_{code}(\sigma_{percentage=100 \wedge Continent='Africa'}(encompasses))$
- (b) $\pi_{lakeName}(riverthrough \bowtie_{river=river1} \rho_{river1 \leftarrow river}(\sigma_{Country='F'}(located)))$
- (c) $\pi_{name}(sea) - \pi_{name}(sea \bowtie_{depth1 > depth} (\rho_{name1, depth1}(sea)))$
- (d) $\rho_{CountryWithTheHighestMountain}(\pi_{name}(\pi_{name}(Mountain) - \pi_{name}(Mountain \bowtie_{elevation < elevation1} \rho_{elevation1 \leftarrow elevation}(Mountain)) \bowtie_{geo_Mountain} Country))$

2. SQL queries

```

1  -- a)
2  SELECT DISTINCT l.country FROM language l
3      WHERE l.name = 'German' OR l.name = 'English';
4
5
6  -- b)
7  SELECT DISTINCT l.name FROM
8      Religion r JOIN Language l ON r.country=l.country
9      WHERE r.name = 'Buddhist';
10
11 -- c)
12 SELECT river FROM River EXCEPT
13 SELECT river FROM encompasses
14     NATURAL INNER JOIN geo_source
15     WHERE continent='Europe';
16

```

```
17 -- d)
18 SELECT DISTINCT c.name, lake, mountain FROM
19     Country c LEFT OUTER JOIN geo_lake l ON c.code=l.country
20         LEFT OUTER JOIN geo_Mountain m ON c.code=m.country
21 WHERE lake IS NOT NULL OR mountain IS NOT NULL;
```