University of Basel

Dr. Ilir Fetai
Prof. Dr. Heiko Schuldt
David Lengweiler, M. Sc.
Martin Vahlensieck, M. Sc.

Database Systems — Autumn 2025

Exercise 5 — Hand-in: 7.12.2025 (11:59 pm)

Solving the Exercises: The exercises must be solved in groups of three people. Use the notations introduced in the lecture. Please choose the same partners for exercises and project. The DMI plagiarism guidelines apply for this lecture.

Submission Information: Please upload your solutions BEFORE the deadline to ADAM using the team hand-in feature. Solutions that are handed in too late cannot be considered. The use of the exercise template, located in "Exercise Material", is mandatory.

Task 1: Query Optimization (13 points)

In complex queries, the ordering of the execution of operations has strong influence on the query time. Given are the relations $U(\underline{a}, b, \utilde{c}), V(\underline{c}, d, \utilde{e})$ and $W(\underline{e}, f, g)$ together with the following SQL statement:

```
SELECT b, f
FROM ( SELECT V.c,V.d,W.e,W.f,W.g
       FROM V, W
       WHERE V.e = W.e
         AND f = 2
         AND g = 10 )
JOIN U ON U.c = V.c
WHERE a LIKE 'Ben';
```

a) Translate the SQL statement to an expression in relational algebra. Then draw the algebraic operator tree for the given SQL statement.

(2 points)

b) Play the query optimizer and transform the operator tree algebraically to the most low-priced operator tree. Perform the transformations first on the algebraic expression and then draw the optimized operator tree. State all the equivalence rules you have applied and in which order.

(7 points)

c) Assume that all attribute values of attributes a, f and g are uniformly and uncorrelated distributed over 10 values. The values of the foreign keys ($U.c$ and $U.e$) are as well distributed uniformly with respect to their key values. The cardinalities of the relations are given by $Card(U) = 10$, $Card(V) = 10'000$, $Card(W) = 100'000$.

Compute the size of the expected intermediate result set (number of tuples) when executing the non-optimized and the optimized operator tree, respectively.

(4 points)

In this exercise you should consider and evaluate various alternatives for the physical layout of a database.

***Note**: Always provide both the calculations and tables containing the final results.*

Data We assume a relation `R`, which contains two numerical attributes A and B. Both attributes have a length of 6 B. In this relation, 50 000 tuples are stored. The attribute values of A are uniformly drawn from the interval $[0 \dots 999]$. The attribute values of B are drawn from the interval $[0 \dots 999]$, whereas 40 000 tuples are uniformly distributed over the interval $[0 \dots 99]$ and 10 000 tuples over the interval $[100 \dots 999]$.

Layout alternatives Various possibilities are at hand for the physical layout of relation `R`. In the course of this exercise, you should consider the following 6 possibilities:

1. There exists only relation `R` (i.e., there are no indices).
2. There exists an index (B^+-tree) on the attribute A (layout name `RA`).
3. There exists an index (B^+-tree) on the attribute B (layout name `RB`).
4. There exists an index on A and a second index on B (B^+-tree, layout name `RAB`).
5. There exists a clustered, direct index on A, i.e., the relation is sorted physically with respect to attribute A (layout name `RA$`).
6. There exists a combined index (B^+-tree) on A and B, i.e., the index key is the concatenation of the attribute values (layout name `RC`).

All indices except alternative 5 are indirect (indirect B^+-tree). Assume that the relations and indices are stored in different table spaces, whereat the page size is 8 KiB with a filling degree of 70 % for the index pages and 90 % for the data pages. A TID or pointer in the B^+-tree is of length 6 B. 48 B per page are used for page information and 6 B for a slot array pointer.

Hint: All queries in this exercise contain only `SELECT COUNT(*)`. Hence, there is no need to access any data pages if an index is used.

Task 2: Page Accesses for Simple Queries (10 points)

Determine the height of the indicies and the number of page accesses necessary for queries of the form `SELECT COUNT(*) FROM R WHERE A=10;`. Use values 10 and 500 for concrete values of A and B.

Task 3: Page Accesses for Complex Queries (7 points)

Determine the number of page accesses necessary for the following query:

Layout	Height	Number of page accesses			
		$A = 10$	$B = 10$	$A = 500$	$B = 500$
R: no index					
RA: index on A					
RB: index on B					
RAB: indices on A and B					
RA$: direct, clustered index					
RC: combined index					

Q:
```
SELECT COUNT(*)
FROM R
WHERE A = 321 AND B <> 50;
```

Layout	Number of page accesses
	Q
R: no index	
RA: index on A	
RB: index on B	
RAB: indices on A and B	
RA$: direct, clustered index	
RC: combined index	