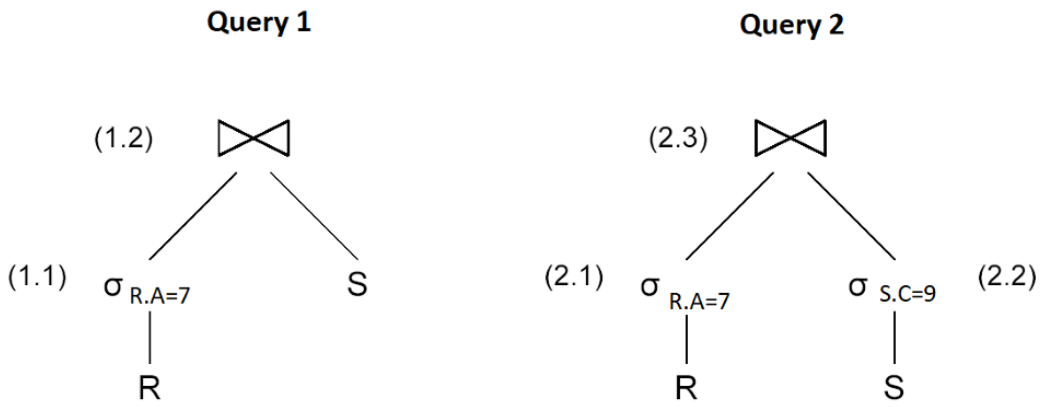
Question 1: Query Optimization with Hash Join

Consider the following queries on relations $R(A, B)$ and $S(B, C)$:

Query 1: SELECT * FROM R, S WHERE R.B = S.B AND R.A = 5

Query 2: SELECT * FROM R, S WHERE R.B = S.B AND R.A = 5 AND S.C = 10

The following two plans are being considered:



Relation R has 50,000 tuples with 20 tuples per block, $V(R, A) = 10$ and $V(R, B) = 15$. Similarly, S has 40,000 tuples with 25 tuples per block, $V(S, B) = 8$ and $V(S, C) = 25$. Assume values are distributed over possible $V(\text{Relation}, \text{Attribute})$ values (not over possible domain values).

In this exercise, we make the following additional assumptions:

- The join is implemented as a hash-join;
- The intermediate result produced by operation (1.1) is not written to disk;
- Hash buckets are stored on disk;
- The final result is kept in main-memory;
- There is enough memory to execute the hash join algorithm.

a) How many I/O's does Query 1 require?

b) Also assume that the result of neither (2.1) nor (2.2) is stored to disk. How many I/O's does Query 2 require?

Q1

$B(R) = \frac{50{,}000}{20} = 2500$ bk

$B(S) = \frac{40{,}000}{25} = 1600$ bk

$T(1.1) = T(2.1) = \frac{T(R)}{V(A,R)} = \frac{50{,}000}{10} = 5000$ tuples

$T(2.2) = \frac{T(S)}{V(S,C)} = \frac{40{,}000}{25} = 1600$ tuples

$B(1.1) = B(2.1) = \frac{250 \text{ tp}}{20 \text{ tp/bk}} = 250$ bk

$B(2.2) = \frac{1600 \text{ tp}}{25 \text{ tp/bk}} = 64$ bk

a) query1: we need 2500 I/O to read R and execute selection.
we need 250 I/O to write the hash to disk and again 250 I/O's to read from disk in second phase.
from S, we need 1600 I/O to read S and create hash function,
we need another 1600 to write these hash buckets to disk and again 1600 I/O's to read from disk in second phase of the hash join:

total [7800 I/O's]

b) Query-2: We already have 3000 I/O from left branch.
we need 1600 I/O to read S and execute selection
we need 64 to write hash to disk and again 64 to read from disk:

in total: [4728 I/O's]

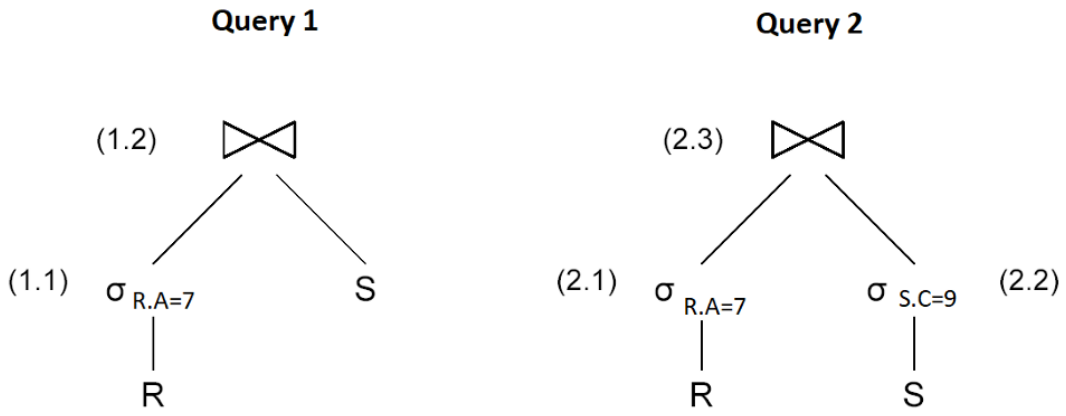
Question 1b: Query Optimization with Hash Join

Consider the following queries on relations $R(A, B)$ and $S(B, C)$:

Query 1: SELECT * FROM R, S WHERE R.B = S.B AND R.A = 10

Query 2: SELECT * FROM R, S WHERE R.B = S.B AND R.A = 10 AND S.C = 20

The following two plans are being considered:



Relation R has 100,000 tuples with 25 tuples per block, $V(R, A) = 12$ and $V(R, B) = 25$.
Similarly, S has 80,000 tuples with 30 tuples per block, $V(S, B) = 15$ and $V(S, C) = 40$.
Assume values are distributed over possible $V(\text{Relation}, \text{Attribute})$ values (not over possible domain values).
In this exercise, we make the following additional assumptions:

- The join is implemented as a hash-join;
- The intermediate result produced by operation (1.1) is not written to disk;
- Hash buckets are stored on disk;
- The final result is kept in main-memory;
- There is enough memory to execute the hash join algorithm.

a) How many I/O's does Query 1 require?
b) Also assume that the result of neither (2.1) nor (2.2) is stored to disk. How many I/O's does Query 2 require?

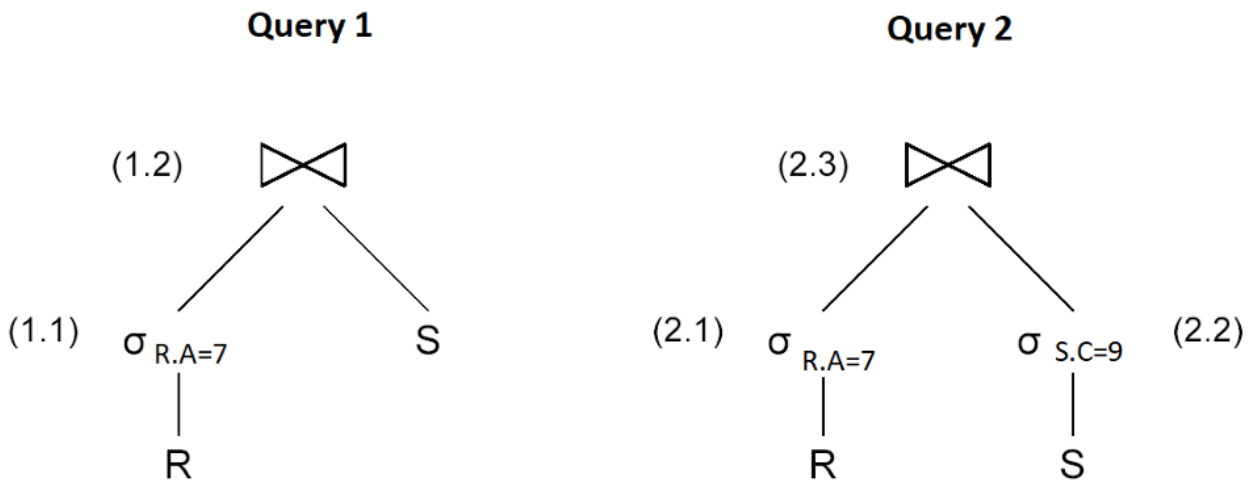
Question 1c: Query Optimization with Different Selectivity

Consider the following queries on relations $R(A, B)$ and $S(B, C)$:

Query 1: SELECT * FROM R, S WHERE R.B = S.B AND R.A = 3

Query 2: SELECT * FROM R, S WHERE R.B = S.B AND R.A = 3 AND S.C = 15

The following two plans are being considered:



Relation R has 70,000 tuples with 15 tuples per block, $V(R, A) = 7$ and $V(R, B) = 20$. Similarly, S has 50,000 tuples with 20 tuples per block, $V(S, B) = 10$ and $V(S, C) = 30$. Assume values are distributed over possible $V(\text{Relation}, \text{Attribute})$ values (not over possible domain values). In this exercise, we make the following additional assumptions:

- The join is implemented as a hash-join;
- The intermediate result produced by operation (1.1) is not written to disk;
- Hash buckets are stored on disk;
- The final result is kept in main-memory;
- There is enough memory to execute the hash join algorithm.

a) How many I/O's does Query 1 require?

b) Also assume that the result of neither (2.1) nor (2.2) is stored to disk. How many I/O's does Query 2 require?

Exercise 1a

Consider the following queries on relations **T(X, Y)** and **U(Y, Z):**

- **Query 1**: `SELECT * FROM T, U WHERE T.Y = U.Y AND T.X = 5`
- **Query 2**: `SELECT * FROM T, U WHERE T.Y = U.Y AND T.X = 5 AND U.Z = 10`

The following two plans are being considered:

Query Plan 1

1. **Filter**: Apply selection **T.X = 5** on **T**
2. **Join**: Perform a hash join with **U** on **T.Y = U.Y**

Query Plan 2

1. **Filter**: Apply selection **T.X = 5** on **T**
2. **Filter**: Apply selection **U.Z = 10** on **U**
3. **Join**: Perform a hash join on **T.Y = U.Y**

Assumptions:

- **T** has **80,000** tuples, **U** has **40,000** tuples
- **T** has **20 tuples per block**, **U** has **40 tuples per block**
- **V(T, Y) = 10**, **V(U, Y) = 8**, **V(U, Z) = 25**
- The join is implemented as a **hash join**
- Intermediate results are **not written to disk**
- There is **enough memory** to execute the join

Questions:

a) How many I/O's does Query 1 require?
b) How many I/O's does Query 2 require?

Exercise 1b

Consider the following queries on relations **A(M, N)** and **B(N, P):**

- **Query 1**: `SELECT * FROM A, B WHERE A.N = B.N AND A.M = 3`
- **Query 2**: `SELECT * FROM A, B WHERE A.N = B.N AND A.M = 3 AND B.P = 7`

Query Plan 1

1. **Filter**: Apply selection **A.M = 3** on **A**
2. **Join**: Perform a hash join with **B** on **A.N = B.N**

Query Plan 2

1. **Filter**: Apply selection **A.M = 3** on **A**
2. **Filter**: Apply selection **B.P = 7** on **B**
3. **Join**: Perform a hash join on **A.N = B.N**

Assumptions:

- **A** has **120,000** tuples, **B** has **60,000** tuples
- **A** has **15 tuples per block**, **B** has **30 tuples per block**

- **V(A, N) = 12**, **V(B, N) = 10**, **V(B, P) = 30**
- The join is implemented as a **hash join**
- Intermediate results are **not written to disk**
- There is **enough memory** to execute the join

Questions:

a) How many I/O's does Query 1 require?
b) How many I/O's does Query 2 require?

Exercise 1c)

Consider the following queries on relations **X(A, B)** and **Y(B, C):**

- **Query 1**: `SELECT * FROM X, Y WHERE X.B = Y.B AND X.A = 2`
- **Query 2**: `SELECT * FROM X, Y WHERE X.B = Y.B AND X.A = 2 AND Y.C = 4`

Query Plan 1

1. **Filter**: Apply selection **X.A = 2** on **X**
2. **Join**: Perform a hash join with **Y** on **X.B = Y.B**

Query Plan 2

1. **Filter**: Apply selection **X.A = 2** on **X**
2. **Filter**: Apply selection **Y.C = 4** on **Y**
3. **Join**: Perform a hash join on **X.B = Y.B**

Assumptions:

- **X** has **90,000** tuples, **Y** has **50,000** tuples
- **X** has **25 tuples per block**, **Y** has **35 tuples per block**
- **V(X, B) = 15**, **V(Y, B) = 12**, **V(Y, C) = 28**
- The join is implemented as a **hash join**
- Intermediate results are **not written to disk**
- There is **enough memory** to execute the join

Questions:

a) How many I/O's does Query 1 require?
b) How many I/O's does Query 2 require?

Question 2a)

Suppose we have two relations **A** and **B** with the following properties:

- **B(A) = 30,000**, **B(B) = 60,000,** The number of available **main memory blocks is M = 150**
- Both relations are **clustered but unsorted**

We want to perform a **merge join** on **A and B** using the available memory.

Questions:

a) How many sorting passes are required before the merge phase?
b) Describe how the merge join is executed and compute the total I/O cost.

Question 2b)

Consider two relations **T(X, Y)** and **U(Y, Z):**

- **B(T) = 25,000**, **B(U) = 75,000, M = 120** main memory blocks are available
- Both relations are **clustered but not sorted**

We aim to join **T and U** on **T.Y = U.Y** using the **merge join algorithm**.

Questions:

a) Determine the number of passes required to sort each relation before merging.
b) Explain the execution of the merge join and calculate the total I/O cost.

Question 2c)

Given the relations **R(A, B)** and **S(B, C)**: **B(R) = 50,000**, **B(S) = 90,000M = 200** main memory blocks

- Both relations are **clustered but initially unsorted,** We need to perform a **merge join** on **R and S**.

Questions:

a) Compute the number of sorting passes needed before merging.
b) Describe the merge join execution and determine the I/O cost.

Question 3a: Join with Duplicate Elimination

Relations $R(a, b)$ and $S(b, c)$ have r and s tuples, respectively. In both relations, attribute b takes values 1, 2, ..., 20 only, with equal probability. We also assume that both relations have duplicates, and on average a tuple that appears at all appears **four times** (i.e., eliminating duplicates reduces the size of R or S by a factor of 4). We wish to compute, without duplicates, those tuples in the natural join of R and S that have $b = 20$. Here are three possible plans for answering this query:

Plan A	Plan B	Plan C
T1 := $\sigma_{b=20}(R)$	T4 := $\sigma_{b=20}(R)$	T7 := $\delta(R)$
T2 := $\delta(S)$	T5 := $\sigma_{b=20}(S)$	T8 := $\sigma_{b=20}(S)$
T3 := T1 JOIN T2	T6 := T4 JOIN T5	T9 := T7 JOIN T8
Ans := $\delta(T3)$	Ans := $\delta(T6)$	Ans := $\delta(T9)$

Provide for each plan a formula for the expected sum of sizes of the intermediate relations depending on the values of r and s.

Question 3b: Join with Specific Attribute Value

Relations $R(a, b)$ and $S(b, c)$ have r and s tuples, respectively. In both relations, attribute b takes values 1, 2, ..., 15 only, with equal probability. We also assume that both relations have duplicates, and on average a tuple that appears at all appears **five times** (i.e., eliminating duplicates reduces the size of R or S by a factor of 5). We wish to compute, without duplicates, those tuples in the natural join of R and S that have $b = 15$. Here are three possible plans for answering this query:

Plan A	Plan B	Plan C
T1 := $\sigma_{b=15}(R)$	T4 := $\sigma_{b=15}(R)$	T7 := $\delta(R)$
T2 := $\delta(S)$	T5 := $\sigma_{b=15}(S)$	T8 := $\sigma_{b=15}(S)$
T3 := T1 JOIN T2	T6 := T4 JOIN T5	T9 := T7 JOIN T8
Ans := $\delta(T3)$	Ans := $\delta(T6)$	Ans := $\delta(T9)$

Provide for each plan a formula for the expected sum of sizes of the intermediate relations depending on the values of r and s.

Question 3c: Join with Different Duplicate Factors

Relations $R(a, b)$ and $S(b, c)$ have r and s tuples, respectively. In both relations, attribute b takes values 1, 2, ..., 25 only, with equal probability. We also assume that both relations have duplicates, and on average a tuple that appears at all appears **two times** in R and **three times** in S (i.e., eliminating duplicates reduces the size of R by a factor of 2 and S by a factor of 3). We wish to compute, without duplicates, those tuples in the natural join of R and S that have $b = 25$. Here are three possible plans for answering this query:

Plan A	Plan B	Plan C
T1 := $\sigma_{b=25}(R)$	T4 := $\sigma_{b=25}(R)$	T7 := $\delta(R)$
T2 := $\delta(S)$	T5 := $\sigma_{b=25}(S)$	T8 := $\sigma_{b=25}(S)$
T3 := T1 JOIN T2	T6 := T4 JOIN T5	T9 := T7 JOIN T8
Ans := $\delta(T3)$	Ans := $\delta(T6)$	Ans := $\delta(T9)$

Provide for each plan a formula for the expected sum of sizes of the intermediate relations depending on the values of r and s.

Explanation of Concepts in the Questions

1. **Natural Join**: Combines tuples from R and S based on a common attribute b.
2. **Duplicate Elimination (δ)**: Removes duplicate tuples from a relation.
3. **Selection (σ)**: Filters tuples based on a condition (e.g., $b = 10$).
4. **Intermediate Relations**: Temporary relations created during query execution (e.g., $T1, T2, T3$).
5. **Expected Sizes**: The average sizes of intermediate relations, calculated based on the number of tuples (r and s) and the duplicate factors.

Question 3a:

Consider two relations R(A, B) and S(B, C) with sizes |R| = 50,000 tuples and |S| = 25,000 tuples. The attribute B has a uniform distribution of values. We aim to compute the natural join R⋈S.

Three different query plans are proposed:

- **Plan X:** First, sort both R and S on B and then perform a merge join.
- **Plan Y:** Use a hash join with a hash table built on the smaller relation S.
- **Plan Z:** Perform a nested loop join with R as the outer relation.

a) For each plan, estimate the number of I/Os required.
b) Which plan is optimal given a memory size of M = 200 pages?

Question 3b:

Given relations R(A, B) and S(B, C), we want to evaluate the following query efficiently:

```
SELECT * FROM R, S WHERE R.B = S.B AND R.A < 100 AND S.C > 50;
```

We consider three alternative query execution plans:

- **Plan A:** Apply selection predicates on R and S before joining.
- **Plan B:** Perform the join first, then apply selection predicates.
- **Plan C:** Use an index join, assuming an index exists on S.B.

a) What are the expected intermediate result sizes for each plan?
b) Which plan minimizes the I/O cost assuming uniform data distribution?

Question 3c:

Suppose we have relations R(A, B) with 80,000 tuples and S(B, C) with 40,000 tuples. The number of unique B values in R and S are 4,000 and 2,000, respectively.

We need to compute the join R⋈S using different strategies:

- **Plan 1:** Hash join, with S as the build relation.
- **Plan 2:** Sort-merge join.
- **Plan 3:** Block nested loop join, with R as the outer relation.

a) Compute the estimated number of disk I/Os required for each plan.
b) If the available memory is limited to 500 pages, which plan is the most efficient?

4a)

Consider two relations R(A, B) and S(B, C) with sizes **B(R) = 30,000** and **B(S) = 40,000** blocks, respectively. The available main memory has **M = 150** blocks. We want to perform a **one-pass join** between R and S.

a) How many memory blocks are required to perform the join in a single pass?
b) If one-pass join is not feasible, what would be the best alternative join method given the memory constraints?

4b)

Given two relations R(X, Y) and S(Y, Z) stored in **B(R) = 10,000** and **B(S) = 20,000** blocks, we have **M = 250** main memory blocks available. Both relations are **unsorted and unclustered**.

a) Can we perform a **one-pass join** with the given memory?
b) If not, how many passes would a **two-pass sort-merge join** require, and what would be the estimated I/O cost?

Question 4c:

Suppose we have two relations R(P, Q) and S(Q, R) stored in **B(R) = 50,000** and **B(S) = 80,000** blocks. The main memory available for the join operation is **M = 300** blocks.

a) What is the minimum number of memory blocks required to perform a **one-pass hash join**?
b) If we were to use a **partitioned hash join**, how many partitions would be required, and how would they be distributed across memory?

Question 5:

2) Consider a clustered relation R(A, B, C, D) that has a clustering index on A and a non-clustering index on each of the other attributes. The relevant parameters are: B(R) = 1000, T(R) = 5000, V(R, A) = 20, V(R, B) = 1000, V(R, C) = 5000, and V(R, D) = 500. Give the best query plan for the following selection and the corresponding number of disk I/O's. You can ignore the cost for accessing the index.

$\sigma_{A=10 \text{ AND } C=2 \text{ AND } D=3}(R)$

Question 5a:

Consider a clustered relation **R(A, B, C, D)** that has a clustering index on **B** and non-clustering indexes on each of the other attributes. The relevant parameters are:

- B(R) = 5000, T(R) = 10000, V(R, A) = 50, V(R, B) = 200, V(R, C) = 1000, and V(R, D) = 4000.

Determine the best query plan and the number of disk I/Os for the following selection:

$$\sigma_{B=5 \text{ AND } C=20}(R)$$

Question 5b:

Given a relation **S(X, Y, Z, W)** with a clustering index on **X** and a non-clustering index on **Z**, the database parameters are:

- **B(S) = 8000**, **T(S) = 40000**, **V(S, X) = 100**, **V(S, Y) = 500**, **V(S, Z) = 2500**, and **V(S, W) = 10000**.

Identify the optimal query execution plan and calculate the number of disk I/Os required to evaluate the query:

$$\sigma_{X=3 \text{ AND } Z=100}(S)$$

Question 5c:

Suppose we have a relation **T(P, Q, R, S, T)** where:

- A **clustering index** exists on **P**,
- A **non-clustering index** exists on **T**,
- **B(T) = 12000**, **T(T) = 60000**, **V(T, P) = 150**, **V(T, Q) = 5000**, **V(T, R) = 1000**, **V(T, S) = 2000**, **V(T, T) = 100**.

Estimate the best way to process the following selection query and compute the corresponding number of disk I/Os:

$$\sigma_{P=50 \text{ AND } T=3}(T)$$
