4. Logical optimization

Logical optimization

- Algebraic Laws
- Improving the logical query plan

Algebraic laws

- Commutative and associative laws
- Laws involving selection
- Laws involving projection
- Laws about joins and product

Commutative and Associative Laws

Operation	Commutativity	Associativity
Cartesian product	$R \times S = S \times R$[1)]	$(R \times S) \times T = R \times (S \times T)$
Natural join	$R \bowtie S = S \bowtie R$[1)]	$(R \bowtie S) \bowtie T = R \bowtie (S \bowtie T)$[2)]
Union	$R \cup S = S \cup R$	$(R \cup S) \cup T = R \cup (S \cup T)$
Intersection	$R \cap S = S \cap R$	$(R \cap S) \cap T = R \cap (S \cap T)$

1) The order of columns changes.
2) Natural join of three relations is performed on attributes which are common for all the relations.

Restricted associativity of theta-join

Commutativity	**Associativity**
$R \underset{\theta}{\bowtie} S = S \underset{\theta}{\bowtie} R$	?

$$R(A,B);\ S(C,D);\ T(E,F)$$

$$R \underset{B=C}{\bowtie} \left(S \underset{D=E}{\bowtie} T\right) = \left(R \underset{B=C}{\bowtie} S\right) \underset{D=E}{\bowtie} T$$

There is no F attribute in S relation!

$$R \underset{A=F}{\bowtie} \left(S \underset{D=E}{\bowtie} T\right) \neq \left(R \underset{A=\boxed{F}}{\bowtie} S\right) \underset{D=E}{\bowtie} T$$

Laws involving selection

1. $\sigma_{C_1 \& C_2}(R) = \sigma_{C_1}(\sigma_{C_2}(R))$
2. $\sigma_{C_1}(\sigma_{C_2}(R)) = \sigma_{C_2}(\sigma_{C_1}(R))$
3. $\sigma_{A<x}\left(R \underset{(A)}{\bowtie} S\right) = \left(\sigma_{A<x}\left(R\right)\right) \underset{(A)}{\bowtie} \left(\sigma_{A<x}\left(S\right)\right)$

Laws involving projection

1. $\pi_{\alpha}(R \bowtie S) = \pi_{\alpha}\left(\pi_{\beta}(R) \bowtie \pi_{\gamma}(S)\right)$
2. $\pi_{\alpha}(R \underset{\theta}{\bowtie} S) = \pi_{\alpha}\left(\pi_{\beta}(R) \underset{\theta}{\bowtie} \pi_{\gamma}(S)\right)$
3. $\pi_{\alpha}(R \times S) = \pi_{\alpha}\left(\pi_{\beta}(R) \times \pi_{\gamma}(S)\right)$

Projection over natural join

$$\pi_{\alpha}(R \bowtie S) = \pi_{\alpha}\left(\pi_{\beta}(R) \bowtie \pi_{\gamma}(S)\right)$$

β - the join attributes and the attributes of α that are found among the attributes of R

γ - the join attributes and the attributes of α that are found among the attributes of S

Projection over theta-join

$$\pi_{\alpha}(R \underset{\theta}{\bowtie} S) = \pi_{\alpha}\left(\pi_{\beta}(R) \underset{\theta}{\bowtie} \pi_{\gamma}(S)\right)$$

β - the join attributes (i.e., those mentioned in condition θ) and the attributes of α that are found among the attributes of *R*

γ - the join attributes (i.e., those mentioned in condition θ) and the attributes of α that are found among the attributes of *S*

Projection over cartesian product

$$\pi_\alpha(R \times S) = \pi_\alpha\Big(\pi_\beta(R) \times \pi_\gamma(S)\Big)$$

β - the attributes of α that are found among the attributes of *R*

γ - the attributes of α that are found among the attributes of *S*

Laws about joins and product

1. $R \underset{\theta}{\bowtie} S = \sigma_{\theta}(R \times S)$

2. $R \underset{(A)}{\bowtie} S = \pi_{R.*,S.*-S.A}\left(\sigma_{R.A=S.A}\left(R \times S\right)\right)$

$$R \underset{C>E*1000}{\bowtie} S = \sigma_{C>E*1000}(R \times S)$$

R

A*	B	C
1	20	600
2	40	300
3	20	150
4	10	300

S

D*	F	E
1	3	0.2
2	1	0.7
3	1	0.5

$R \underset{C>E*1000}{\bowtie} S$

A*	B	C	D*	F	E
1	20	600	1	3	0.2
1	20	600	3	1	0.5
2	40	300	1	3	0.2
4	10	300	1	3	0.2

1) $Q = R \times S$

A*	B	C	D*	F	E
1	20	600	1	3	0.2
1	20	600	2	1	0.7
1	20	600	3	1	0.5
2	40	300	1	3	0.2
2	40	300	2	1	0.7
2	40	300	3	1	0.5
3	20	150	1	3	0.2
3	20	150	2	1	0.7
3	20	150	3	1	0.5
4	10	300	1	3	0.2
4	10	300	2	1	0.7
4	10	300	3	1	0.5

2) $\sigma_{C>E*1000}(Q)$

A*	B	C	D*	F	E
1	20	600	1	3	0.2
1	20	600	3	1	0.5
2	40	300	1	3	0.2
4	10	300	1	3	0.2

$$R \underset{(A)}{\bowtie} S = \pi_{R.*,S.*-S.A}\left(\sigma_{R.A=S.A}\left(R \times S\right)\right)$$

R

A*	B	C
1	20	100
2	40	300
3	20	100
4	10	300

S

D*	A#	E
1	3	0.2
2	1	0.5
3	1	0.5

W=R⋈S

A*	B	C	D*	E
1	20	100	2	0.5
1	20	100	3	0.5
3	20	100	1	0.2

1) $Q = R \times S$

R.A*	B	C	D*	S.A#	E
1	20	100	1	3	0.2
1	20	100	2	1	0.5
1	20	100	3	1	0.5
2	40	300	1	3	0.2
2	40	300	2	1	0.5
2	40	300	3	1	0.5
3	20	100	1	3	0.2
3	20	100	2	1	0.5
3	20	100	3	1	0.5
4	10	300	1	3	0.2
4	10	300	2	1	0.5
4	10	300	3	1	0.5

2) $P = \sigma_{R.A=S.A}(Q)$

R.A*	B	C	D*	S.A#	E
1	20	100	2	1	0.5
1	20	100	3	1	0.5
3	20	100	1	3	0.2

3) W=$\pi_{R.*,S.*-S.A}(P)$

A*	B	C	D*	E
1	20	100	2	0.5
1	20	100	3	0.5
3	20	100	1	0.2

Improving the logical query plan

- Optimization with selection
- Optimization with projection
- Optimization with duplicate eliminations
- Optimization by composing the selection and cartesian product

Optimization with selection

- Selections can be pushed down the tree as far as they can go (it reduces the size of intermediate relations and may therefore be beneficial).
- If a selection condition is the AND of several conditions, then we can split the condition and push each piece down the tree separately.

Optimization with projection

- Projections can be pushed down the tree (it reduces the size of intermediate relations and may therefore be beneficial).
- New projections can be added.

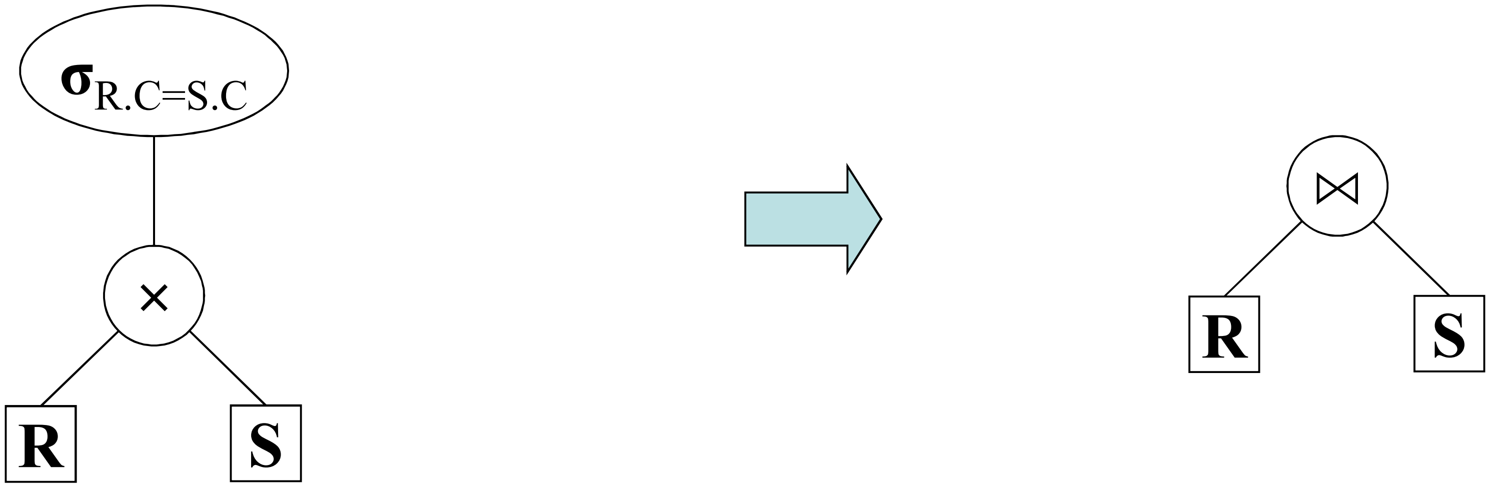
Optimization with duplicate eliminations

- Duplicate eliminations can be pushed down the tree as far as they can go (it reduces the size of intermediate relations and may therefore be beneficial).
- Redundant duplicate eliminations can be eliminated. Relation that is known not to have duplicates :
  - A stored relation with a declared primary key
  - The result of a γ operation, since grouping creates a relation with no duplicates

Optimization by composing the selection and cartesian product

The selection having equality as a condition can be combined with a product below to turn into an equijoin, which is generally much more efficient to evaluate than are the two operations separately.

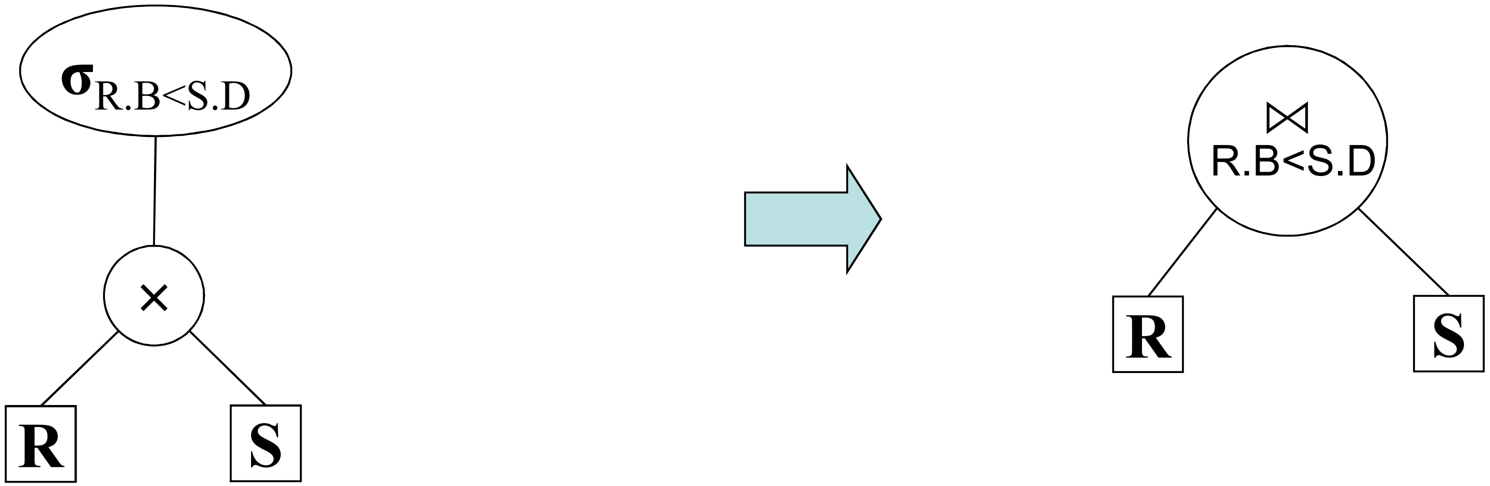
Database schema:
R(A,C); S(C,D)



Optimization by composing the selection and cartesian product

The selection having inequality as a condition can be combined with a product below to turn into an theta-join, which is generally much more efficient to evaluate than are the two operations separately.

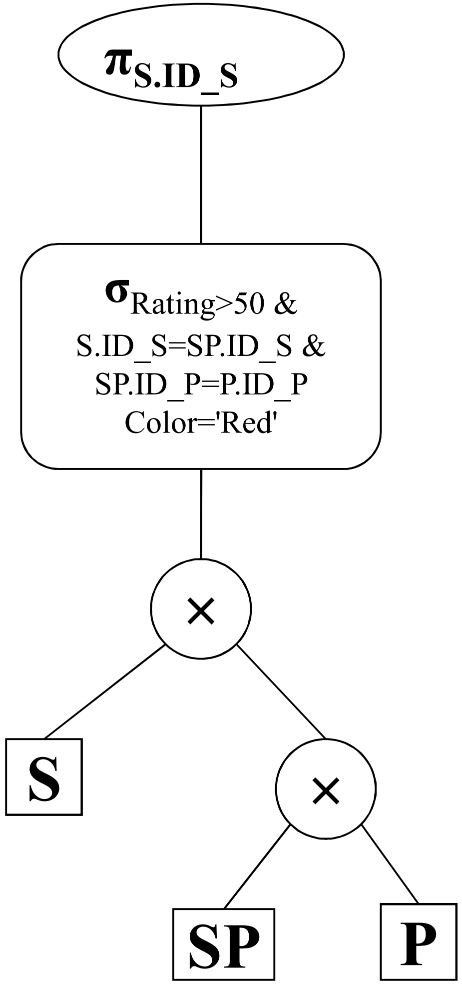
Database schema:
R(A,B); S(C,D)



Example of logical optimization

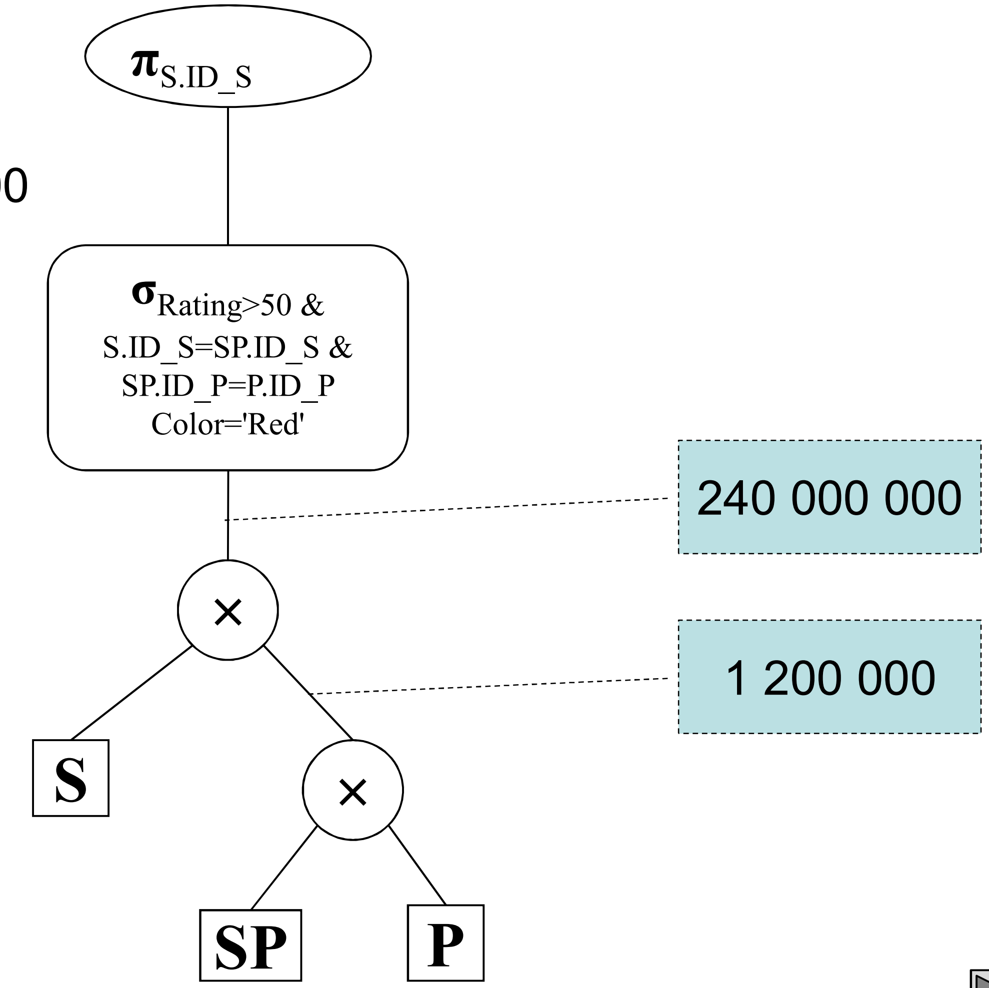
// IDs of suppliers which have rating greater than 50 and supply red parts.

SELECT
ID_S
FROM
S, SP, P
WHERE
Rating > 50 **AND**
S.ID_S = SP.ID_S **AND**
SP.ID_P = P.ID_P **AND**
Color = ‘Red’

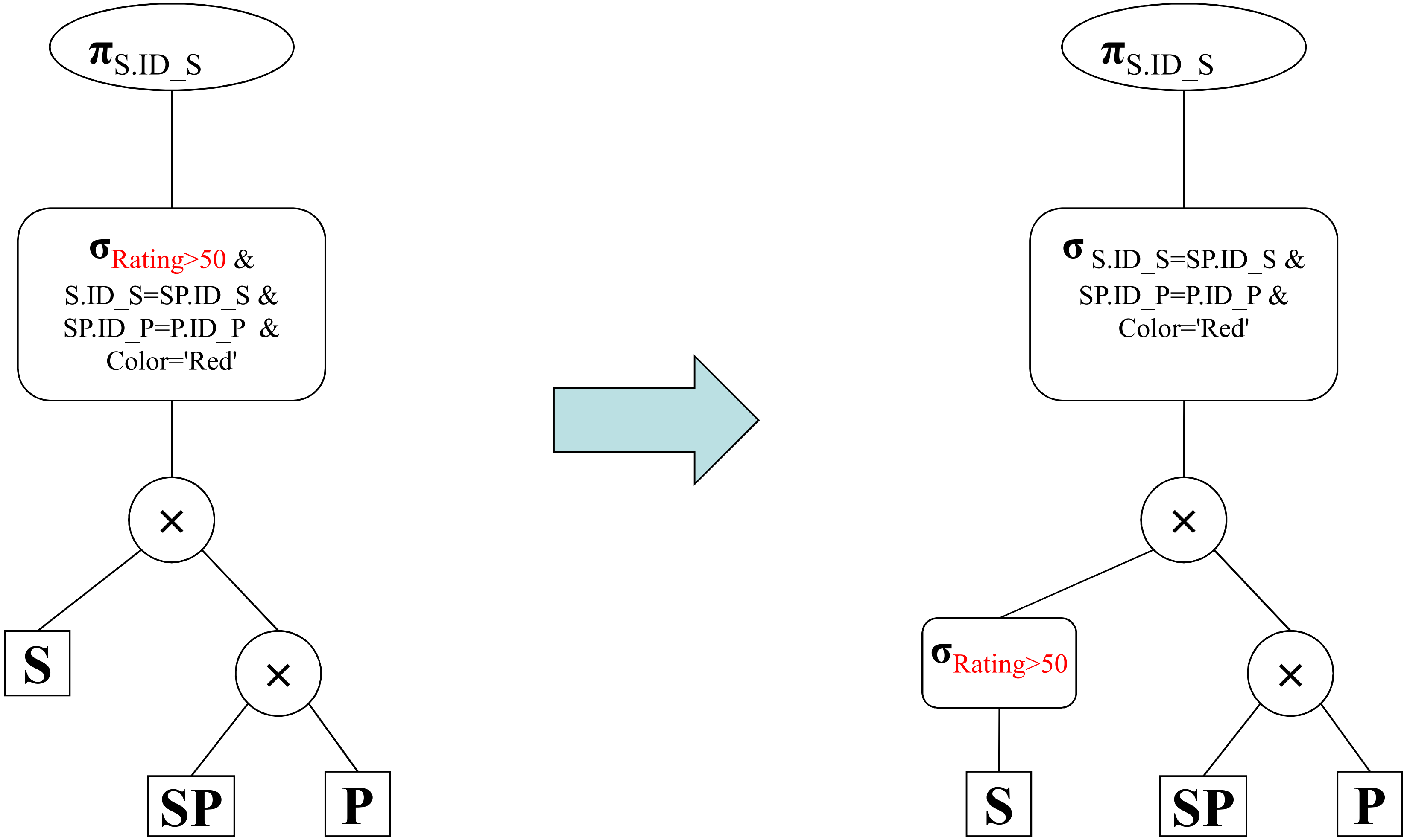


Size of intermediate relations

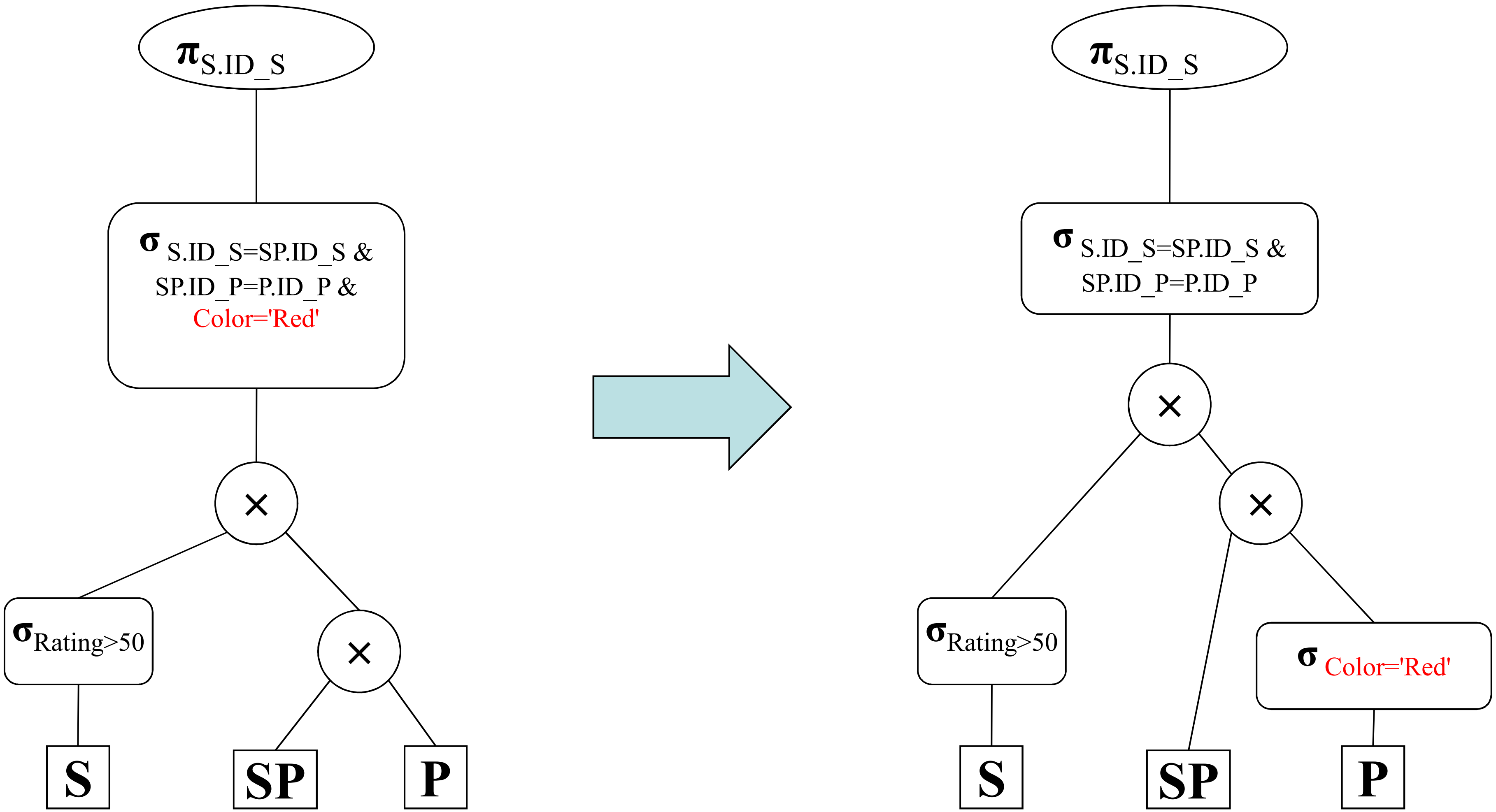
S:	200 tuples
SP:	3000 tuples
P:	400 tuples
SP x P:	3000*400 = 1 200 000
S x SP x P:	1200000*200 = 240 000 000



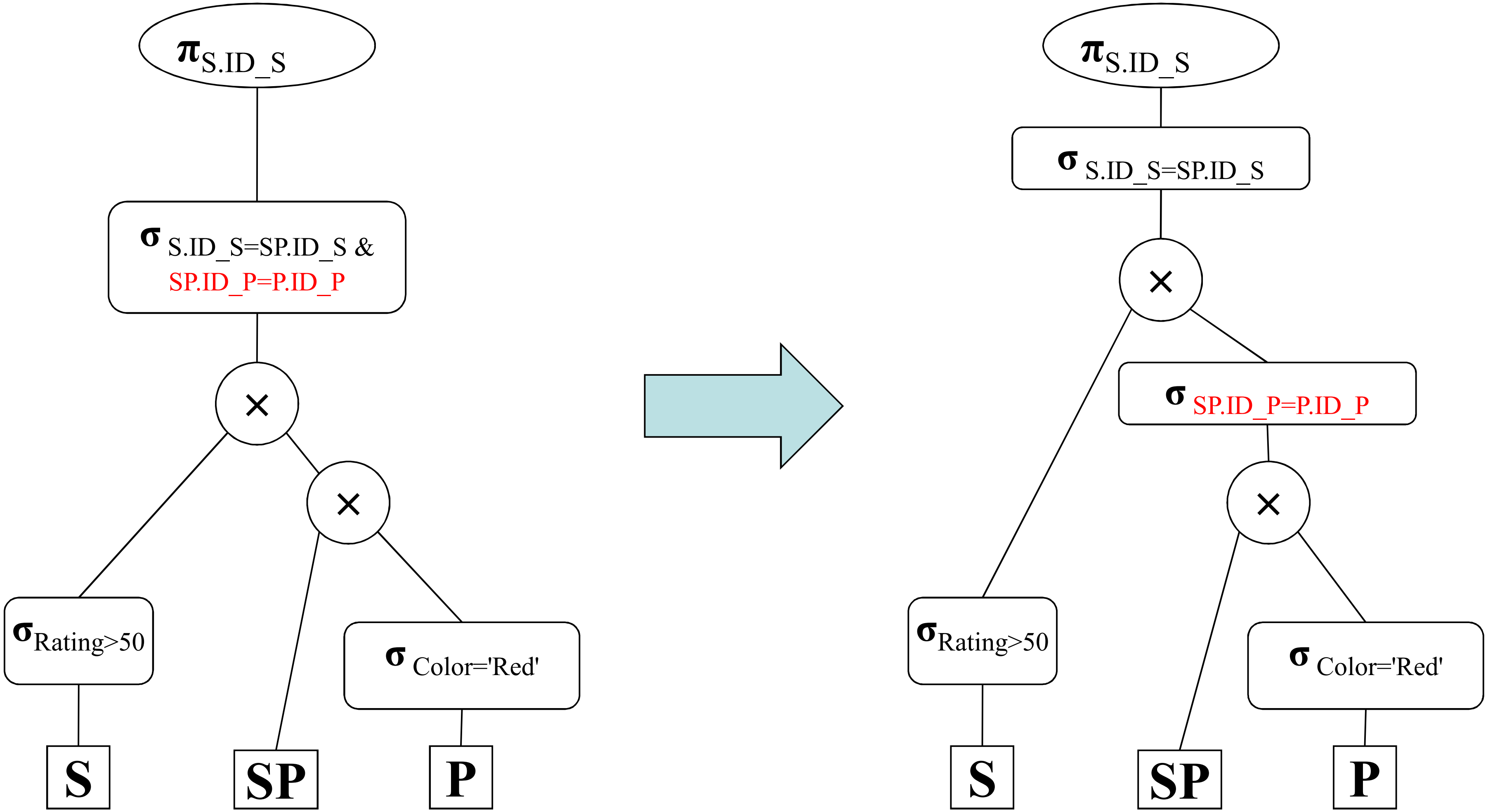
Pushing selections down the tree



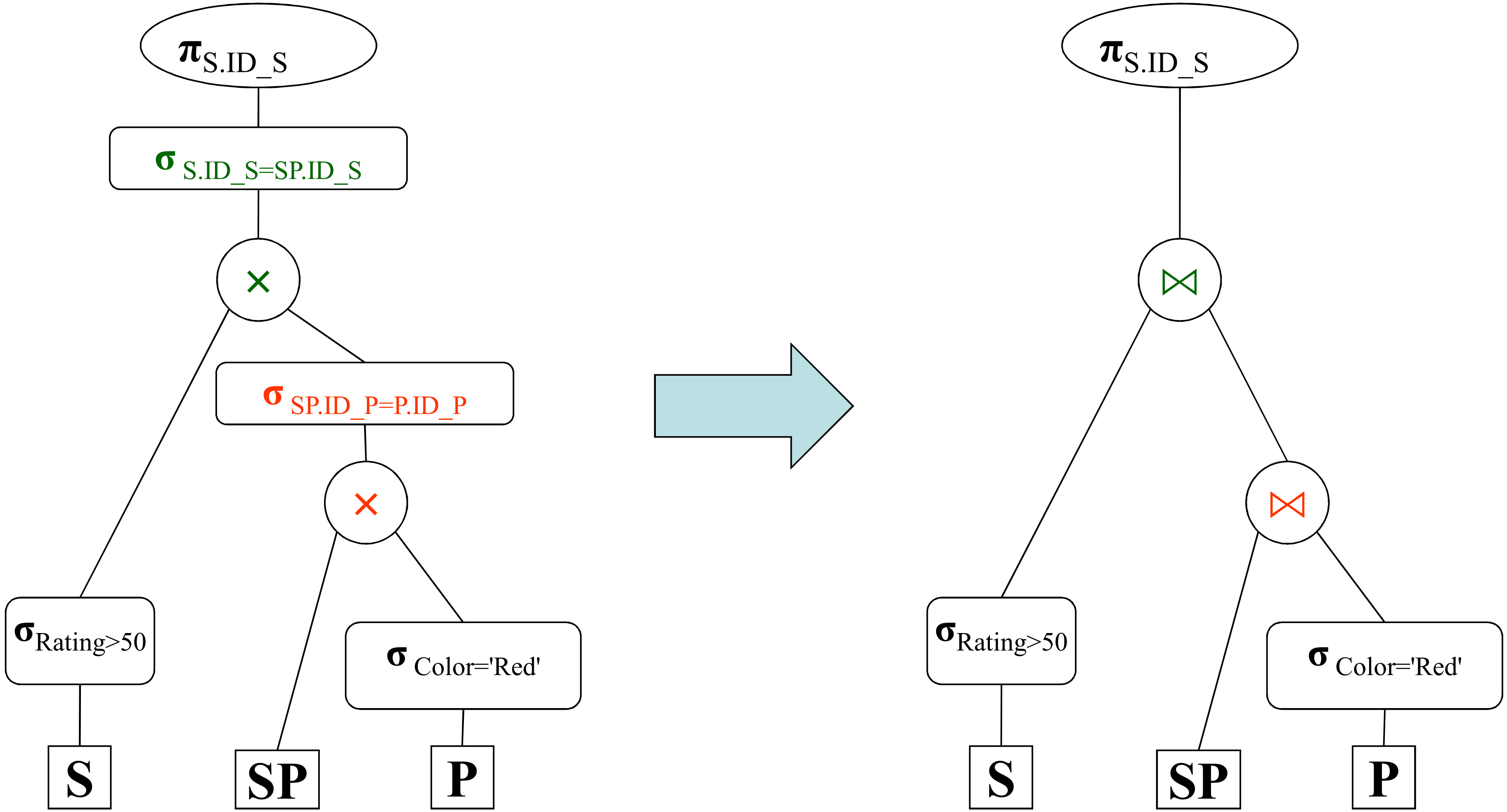
Pushing selections down the tree



Pushing selections down the tree

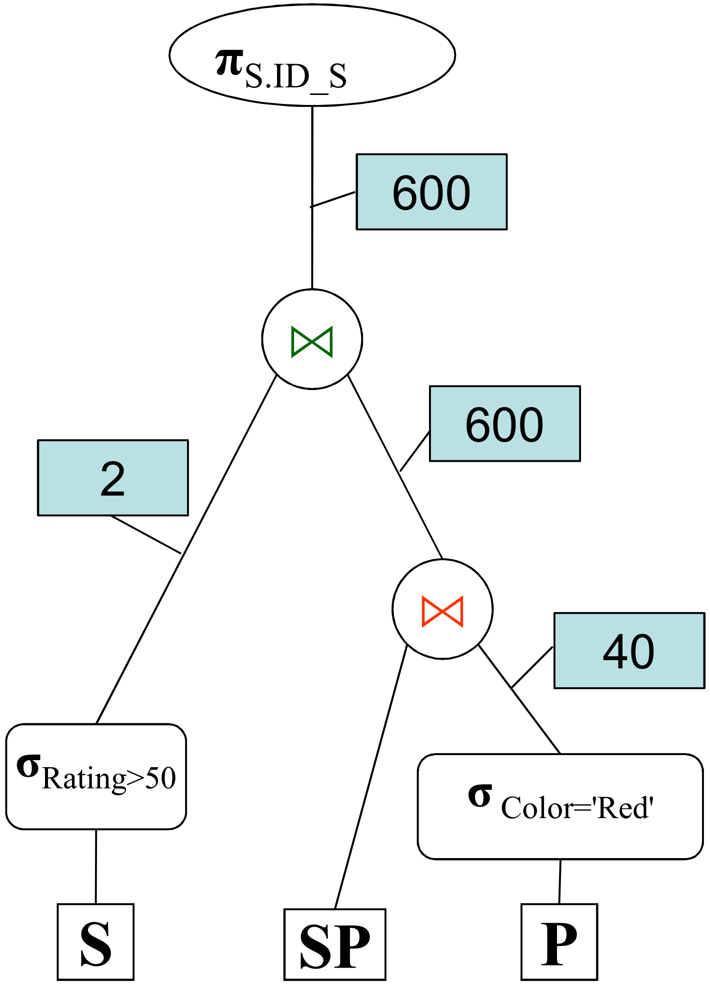


Replacement of selection and cartesian product by equijoin



Size of intermediate relations

S:	200 tuples
SP:	3000 tuples
P:	400 tuples
SP x P:	3000*400 = 1 200 000
S x SP x P:	1200000*200 = 240 000 000



- Red parts: 10%
- Suppliers with rating > 50: 1%
- Supplies of red parts: 20%
- Suppliers supplying red parts: 50%

Pushing projections down the tree