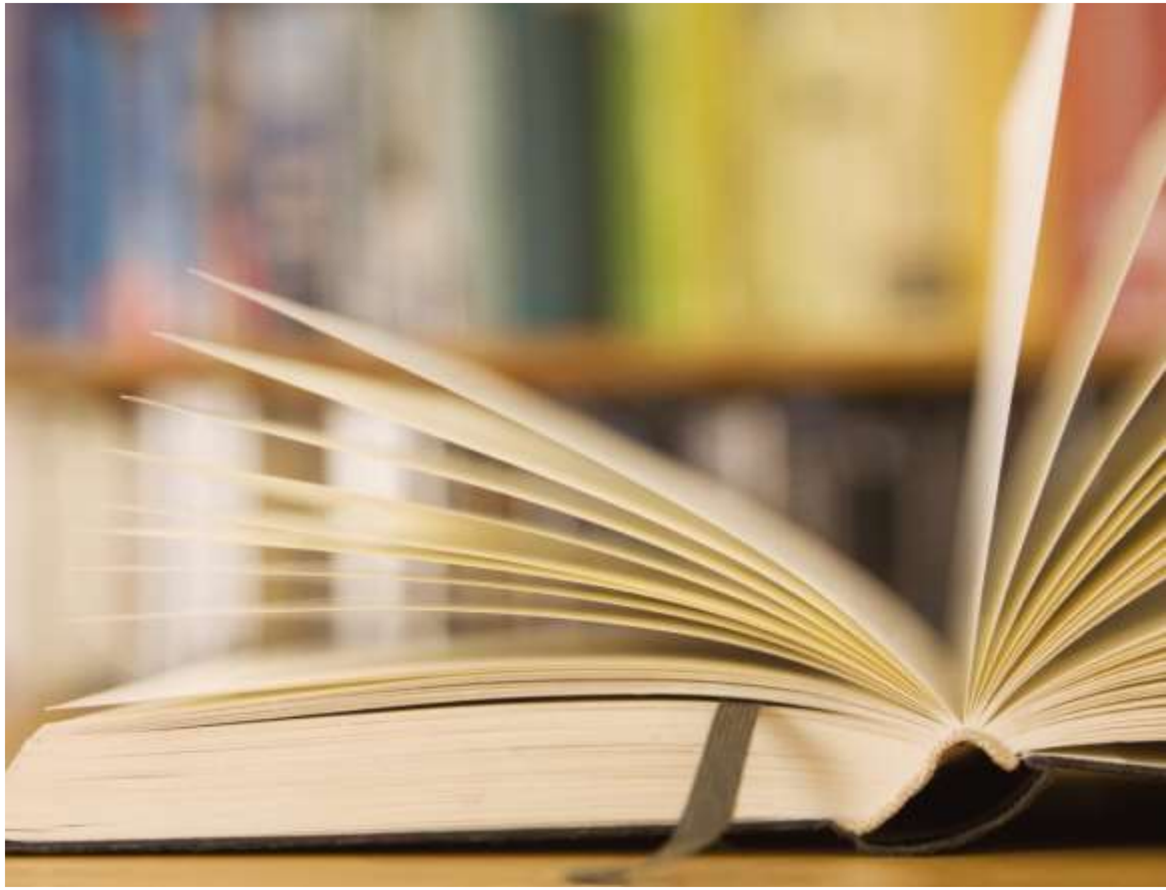
CSCI317 – Database Performance Tuning

Materialized View – Selection Algorithm

17 August 2022



Materialized view …1/3

- Materialized view:
  - A view that is physically stored in a database
  - Materialized views enhance query performance by pre-calculating costly operations such as joins and aggregation and storing the results in the database.

- Materialized view improve query performance at the expense of storage space.

Materialized view …2/3

- A typical problem of materialized views is updating because all modifications to the underlying base tables must be propagated into the view.
- To minimized updating cost**, incremental view maintenance** is used, that is, updated view is computed (done) from changes and not the entire relation.

Materialized view …3/3

- Selection of materialized views is important.
- The goal is to select an appropriate set of views that minimizes the total query response time and the cost of maintaining the selected views, given a limited amount of resources such as storage space or materialization time.

View Selection Algorithm

- Greedy algorithm by Harinarayan et al. (1996)
- Given a cube lattice, finds the best set of views to materialize under a certain criteria.
- It uses a lattice that takes into account two kinds of dependencies between nodes:
  1. **Attributes inclusion**: {ProductKey} INCLUDED-IN {ProductKey, CustomerKey}
  2. **Hierarchy dependency**: Given a hierarchy Month $\rightarrow$ Year, an aggregation over Month can be used to compute the aggregation over Year.

View Selection Algorithm

- Dependency lattice represents a relation $V_i \leq V_j$ such that view V_i can be answered using V_j.
- Algorithm is based on calculating the costs of computing the views in the lattice.
- The view selection algorithm is based on calculating the costs of computing the views in the lattice. A linear cost model with the following characteristics is assumed:
  - The cost of answering a view v from a materialized view v_m is the number of rows in v_m.
  - All queries are identical to some view in the dependency lattice.
  - Requires knowing the expected number of rows for each view in the lattice.
  - The lowest node in the lattice (the base fact table) is always materialized.

View Selection Algorithm

- The goal of the algorithm is to minimize the time taken to evaluate a view, constrained to materialize a fixed number of views regardless of the space available, a problem known as the NP-complete.
- The algorithm works as follow:
- Let $C(v)$ the cost of view v,
- k the number of views to materialize, and
- S a set of materialized views.
- The benefit of materializing a view v not in S, relative to the materialized views already in S, is denoted $B(v, S)$

View Selection Algorithm

ALGORITHM

```
INPUT: A lattice L, each view node labeled with expected number of rows
       A node v, not yet selected to materialize
       A set S containing the nodes already selected to materialize
OUTPUT: The benefit of materializing v given S
BEGIN
  For each view w <= v, B_w is computed as:
      a. Let u be the view of least cost such that w <= u
      b. If C(v) < C(u) , B_w = C(v) - C(u), otherwise B_w = 0
  B(v,S) = SUM_(w<=v) B_w
END
```

View Selection Algorithm

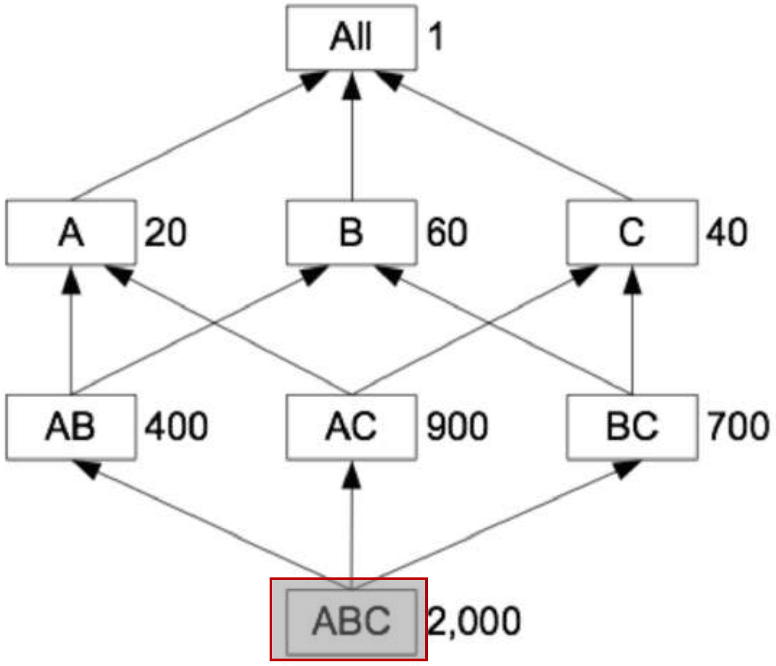
- In each iteration, the algorithm computes the benefit of materializing a view v considering how this improves the cost of computing all the views in the lattice.

ALGORITHM

```
INPUT: A lattice L, each view node v labeled with expected number of rows
OUTPUT: The set of views to materialize
BEGIN
  S = { The bottom view in L}
  FOR i = 1 TO k DO
    Select a view v not in S such that B(v,S) is maximized
    S = S U {v}
  END DO
  S is the selection of views to materialize
END
```

View Selection Algorithm

- For example, based on the dependency lattice shown below, the bottom view (ABC) is already materialized, find another three views to materialize.



Next to each node, the cost of the view the node represents is indicated.

Dependency lattice, with the estimated number of tuples in each node (view)

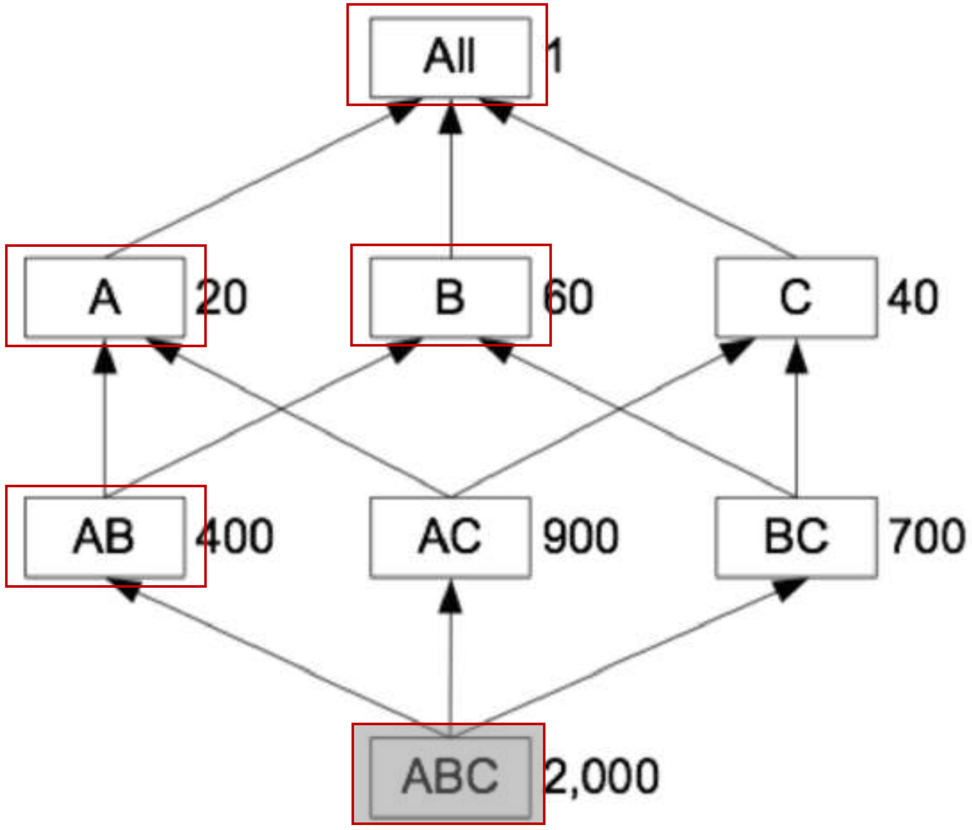
S = {	ABC	}						
ViewToMat = w {	AB,	AC,	BC,	A,	B,	C,	All	}
CostReduction =	-	-	-	-	-	-	-	

To materialized view AB given S, we compute the benefit $B(AB, S) = \sum_{w \leq AB} B_w \, , where\ w = ViewToMat$

For each view w covered by AB, we compute $C(ABC) - C(AB)$, because ABC is the only materialized view at the moment.

Hence, the cost reduction = $C(ABC) - C(AB) = 2000 - 400 = 1600.$

$B(AB, S) = \sum_{w \leq AB} B_w = 1600 + 1600 + 1600 + 1600 = 6400$



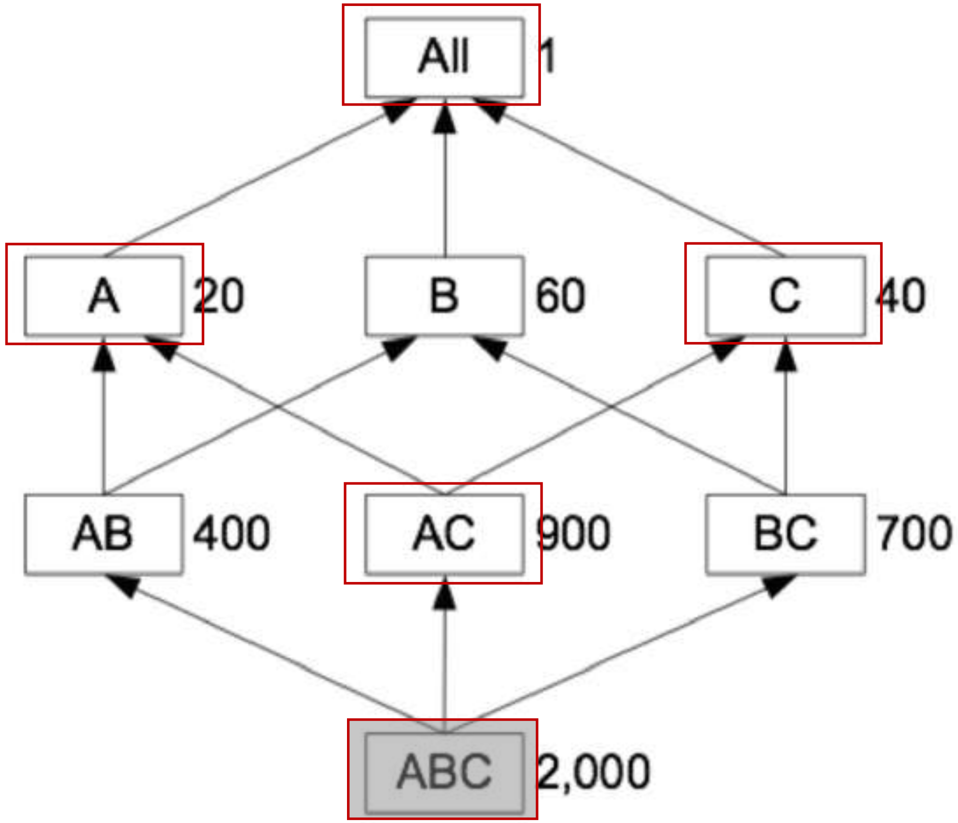
S = {	ABC	}						
ViewToMat = w {	AB,	AC,	BC,	A,	B,	C,	All	}
CostReduction =	6400	-	-	-	-	-	-	

To materialized view AC given S, we compute the benefit $B(AC, S) = \sum_{w \leq AC} B_w, where\ w = ViewToMat$

For each view w covered by AC, we compute $C(ABC) - C(AC)$, because ABC is the only materialized view at the moment.

Hence, the cost $C(ABC) - C(AC) = 2000 - 900 = 1100$.

$B(AC, S) = \sum_{w \leq AC} B_w = 1100 + 1100 + 1100 + 1100 = 4400$



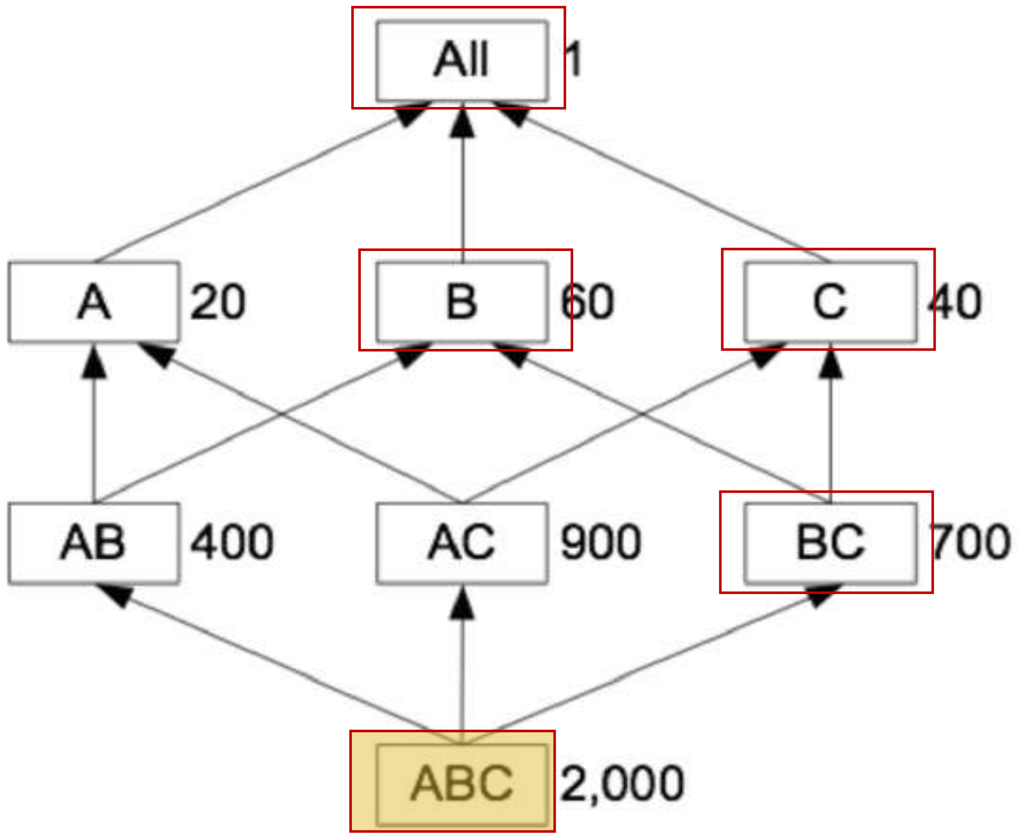
S = {	ABC	}						
ViewToMat = w {	AB,	AC,	BC,	A,	B,	C,	All	}
CostReduction =	6400	4400	-	-	-	-	-	

To materialized view BC given S, we compute the benefit $B(BC, S) = \sum_{w \leq BC} B_w$, $where\ w = ViewToMat$

For each view w covered by BC, we compute $C(ABC) - C(BC)$, because ABC is the only materialized view at the moment.

Hence, the cost $C(ABC) - C(BC) = 2000 - 700 = 1300$.

$B(BC, S) = \sum_{w \leq BC} B_w = 1300 + 1300 + 1300 + 1300 = 5200$



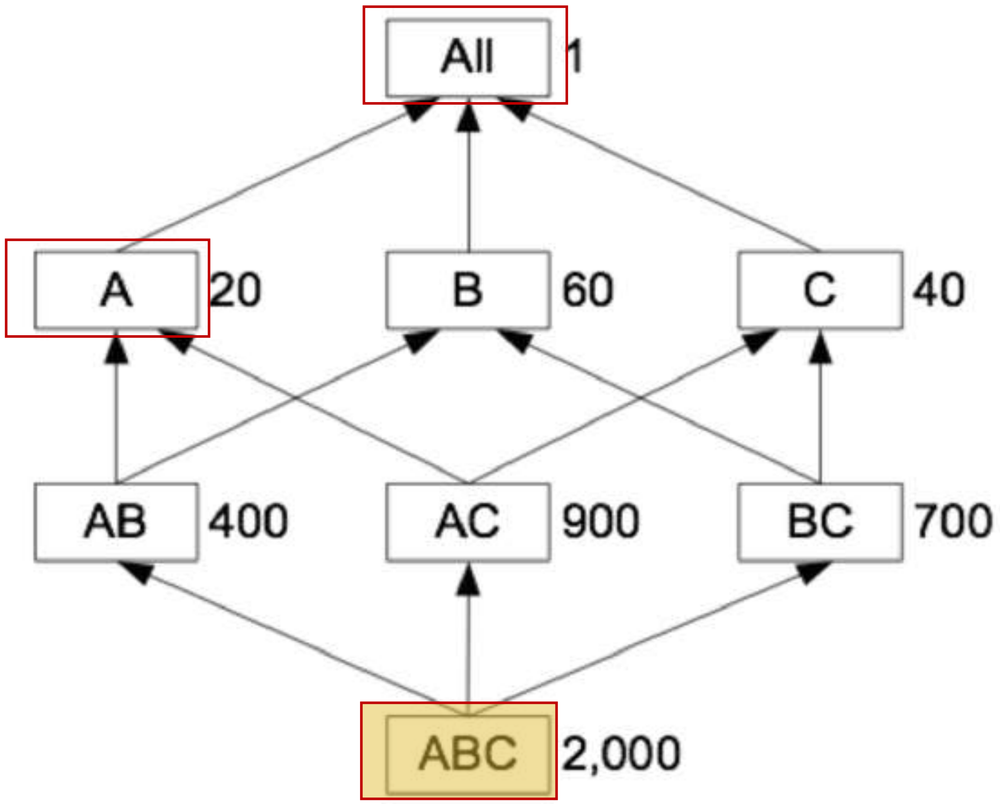
S = {	ABC	}						
ViewToMat = w {	AB,	AC,	BC,	A,	B,	C,	All	}
CostReduction =	6400	4400	5200	-	-	-	-	

To materialized view A given S, we compute the benefit $B(A,S) = \sum_{w \leq A} B_w\,, where\ w = ViewToMat$

For each view w covered by A, we compute $C(ABC) - C(A),)$, because ABC is the only materialized view at the moment.

Hence, the cost $C(ABC) - C(A) = 2000 - 20 = 1980.$

$B(A,S) = \sum_{w \leq A} B_w = 1980 + 1980 = 3960$



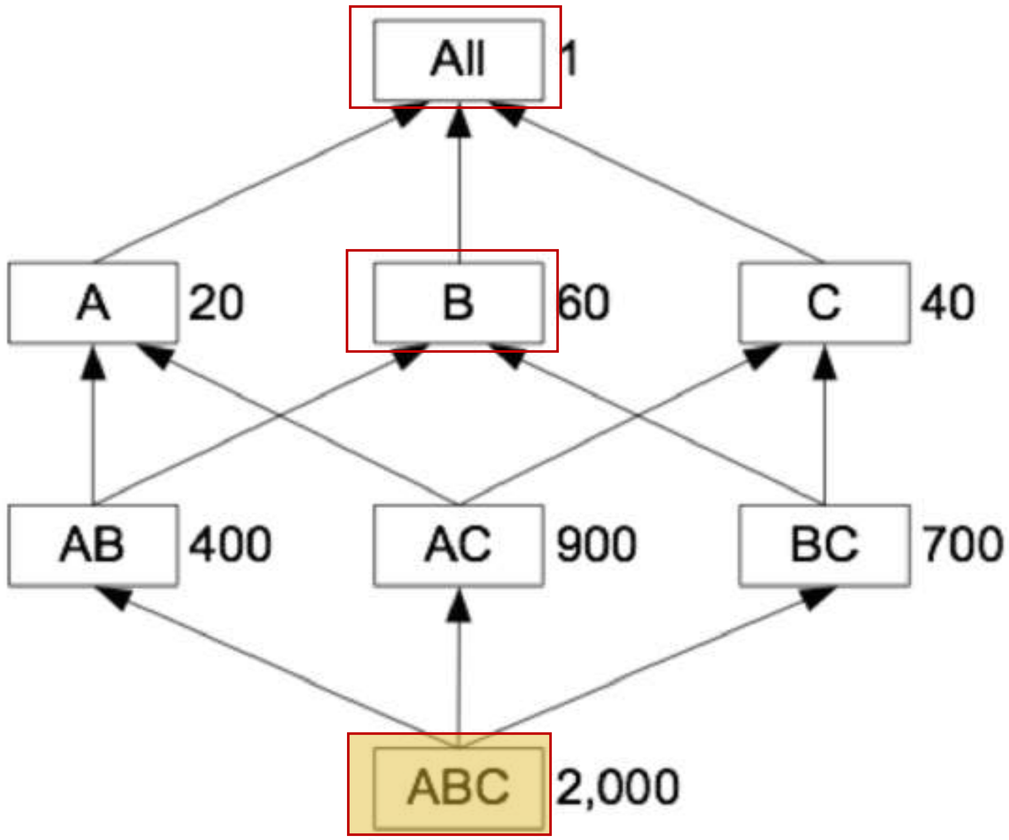
S = {	ABC	}						
ViewToMat = w {	AB,	AC,	BC,	A,	B,	C,	All	}
CostReduction =	6400	4400	5200	3960	-	-	-	

To materialized view B given S, we compute the benefit $B(B,S) = \sum_{w \leq B} B_w \, , where \; w = ViewToMat$

For each view w covered by B, we compute $C(ABC) - C(B),)$, because ABC is the only materialized view at the moment.

Hence, the cost $C(ABC) - C(B) = 2000 - 60 = 1940.$

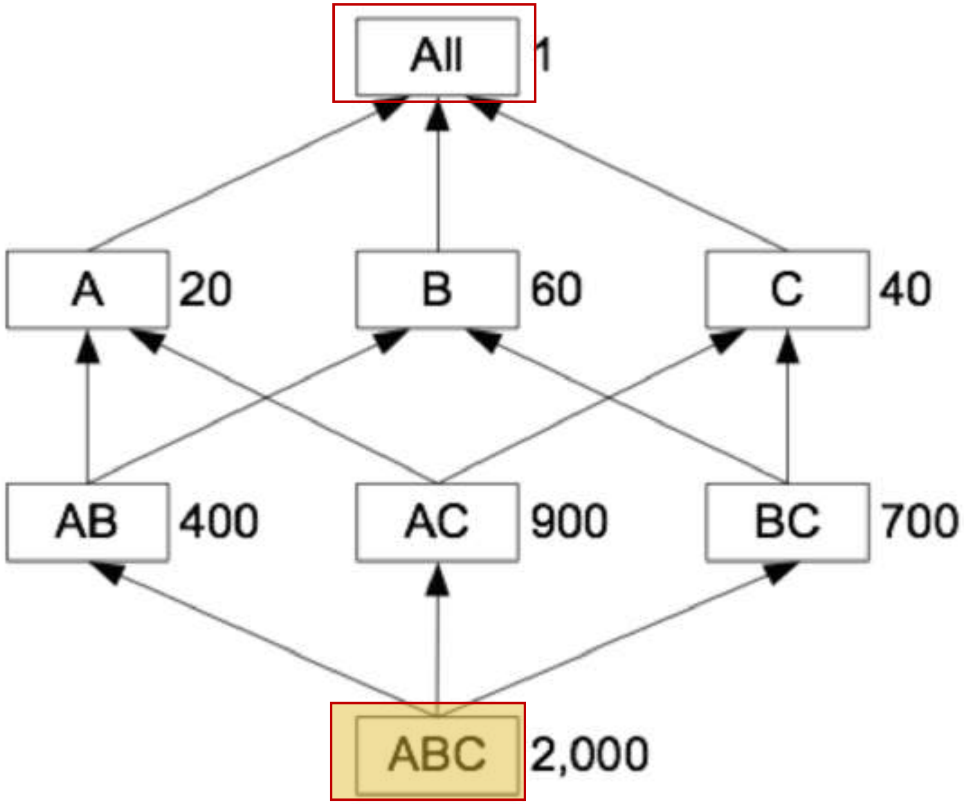
$B(B,S) = \sum_{w \leq B} B_w = 1940 + 1940 = 3880$



S = {	**ABC**	**}**						
ViewToMat = w {	AB,	AC,	BC,	A,	B,	C,	All	}
CostReduction =	6400	4400	5200	3960	3880	3920	1999	

Similar to all previous calculation, the cost reduction to materialized views C and All can be computed and they are 3920 and 1999 respectively.

From the cost reduction obtained, it is noted that 6400 units is the most benefit can be obtained if view AB is materialized. Hence according to the greedy algorithm, the view AB will be materialized.



Second iteration

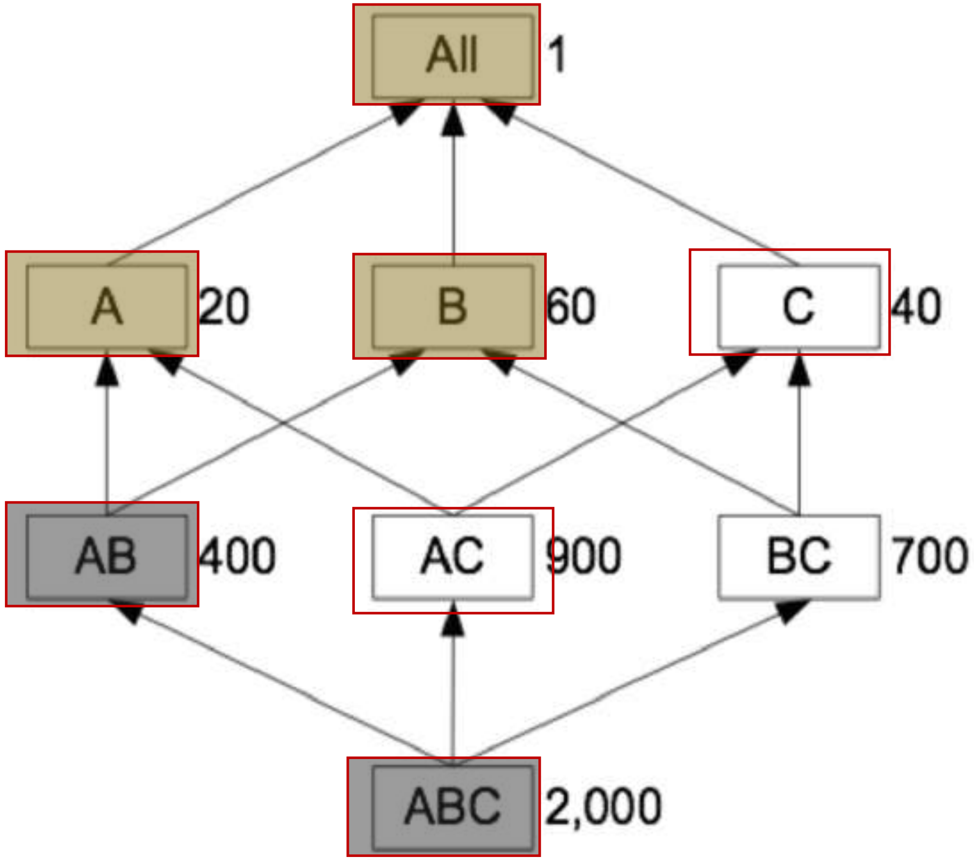
S = {	ABC,	AB	}					
ViewToMat = w {	AB,	AC,	BC,	A,	B,	C,	All	}
CostReduction =	6400	-	-	-	-	-	-	

To materialized view AC given S, we compute the benefit $B(AC, S) = \sum_{w \leq AC} B_w, where\ w = ViewToMat$

For each view w covered by AC, we compute $C(ABC) - C(AC)$, because ABC is dominated by AC.

Hence, the cost reduction = $C(ABC) - C(AC) = 2000 - 900 = 1100.$

$B(AC, S) = \sum_{w \leq AC} B_w = 1100 + 1100 = 2200$



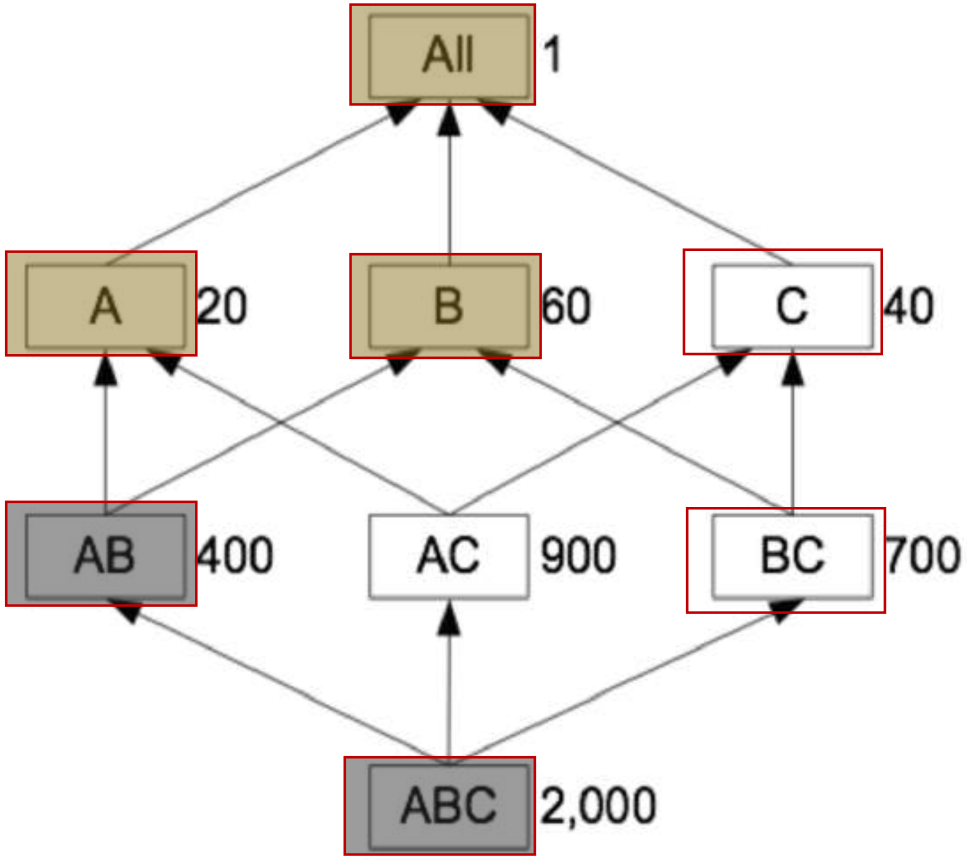
S = {	ABC,	AB	}					
ViewToMat = w {	AB,	AC,	BC,	A,	B,	C,	All	}
CostReduction =	6400	2200	-	-	-	-	-	

To materialized view BC given S, we compute the benefit $B(BC, S) = \sum_{w \leq BC} B_w\, , where\ w = ViewToMat$

For each view w covered by BC, we compute $C(ABC) - C(BC)$, because ABC is dominated by BC.

Hence, the cost reduction = $C(ABC) - C(BC) = 2000 - 700 = 1300$.

$B(BC, S) = \sum_{w \leq BC} B_w = 1300 + 1300 = 2600$



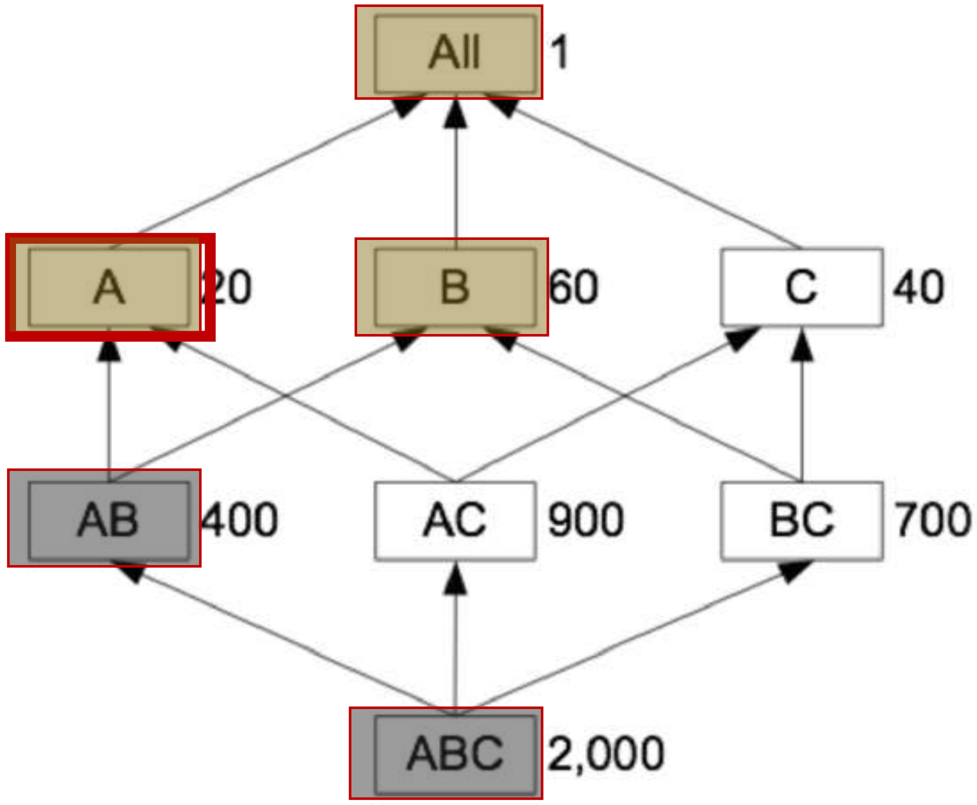
S = {	ABC,	AB	}					
ViewToMat = w {	AB,	AC,	BC,	A,	B,	C,	All	}
CostReduction =	6400	2200	2600	-	-	-	-	

To materialized view A given S, we compute the benefit $B(A,S) = \sum_{w \leq A} B_w \, , where \; w = ViewToMat$

For each view w covered by A, we compute $C(AB) - C(A)$, because AB is dominated by A.

Hence, the cost reduction = $C(AB) - C(A) = 400 - 20 = 380.$

$B(A,S) = \sum_{w \leq A} B_w = 380 + 380 = 760$



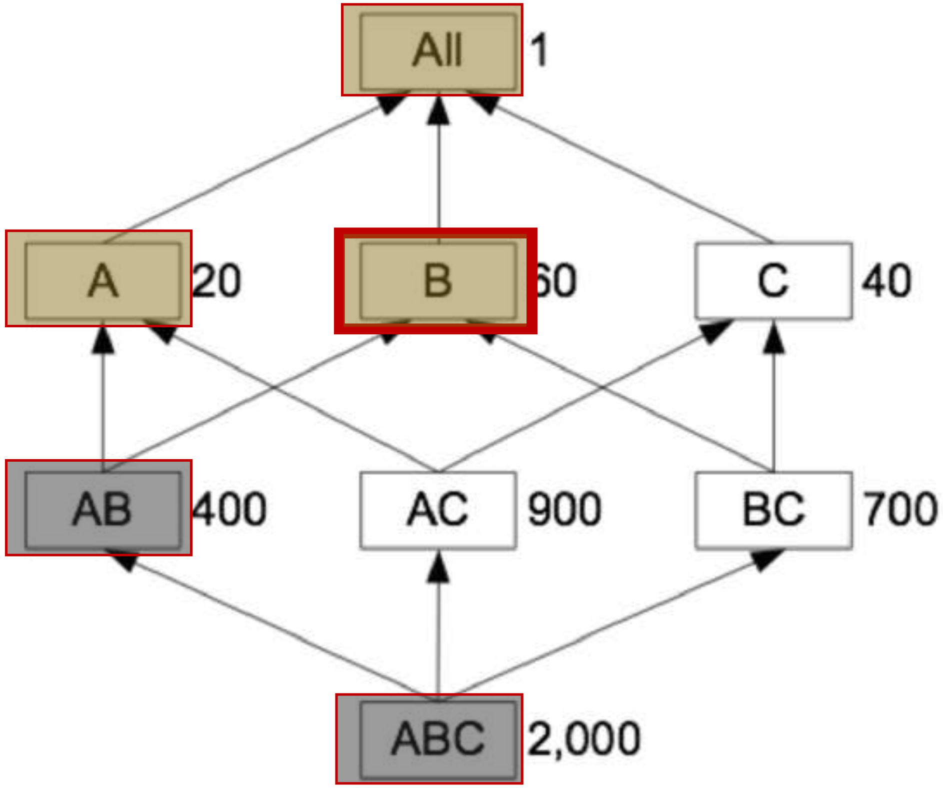
S = {	ABC,	AB	}					
ViewToMat = w {	AB,	AC,	BC,	A,	B,	C,	All	}
CostReduction =	6400	2200	2600	760	-	-	-	

To materialized view B given S, we compute the benefit $B(B,S) = \sum_{w \leq B} B_w\,, where\ w = ViewToMat$

For each view w covered by B, we compute $C(AB) - C(B)$, because AB is dominated by B.

Hence, the cost reduction = $C(AB) - C(B) = 400 - 60 = 340.$

$B(B,S) = \sum_{w \leq B} B_w = 340 + 340 = 680$



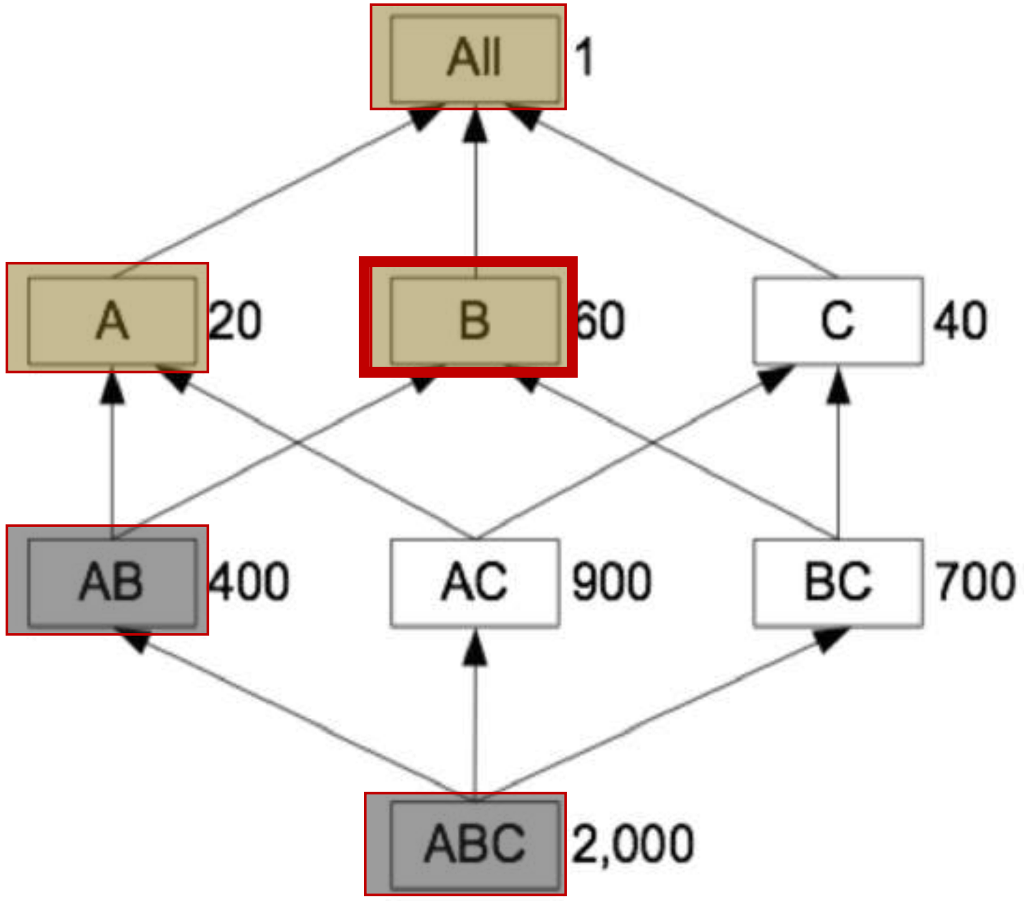
S = {	ABC,	AB	}					
ViewToMat = w {	AB,	AC,	BC,	A,	B,	C,	All	}
CostReduction =	6400	2200	2600	760	680	-	-	

To materialized view C given S, we compute the benefit $B(C,S) = \sum_{w \leq C} B_w\,, where\ w = ViewToMat$

For each view w covered by C, we compute $C(ABC) - C(C) + C(AB) - C(C)$, because ABC is dominated by C and AB is in S.

Hence, the cost reduction = $C(ABC) - C(C) + C(AB) - C(C) = (2000 - 40) + (400 - 40) = 2320.$

$B(C,S) = \sum_{w \leq c} B_w = 2320$



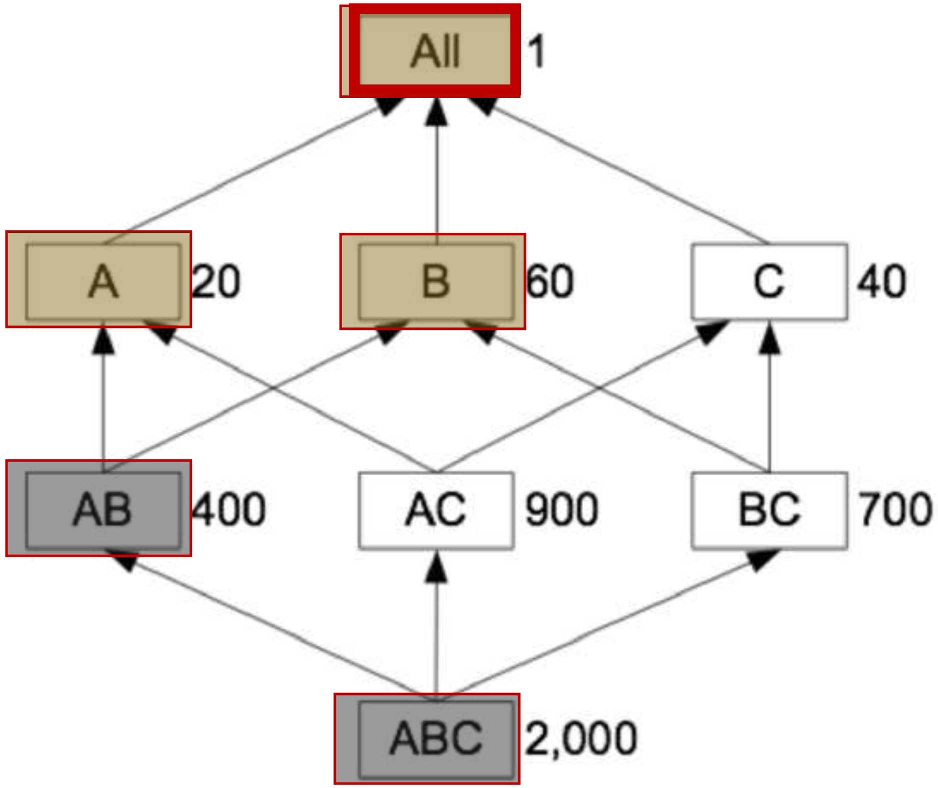
S = {	ABC,	AB	}					
ViewToMat = w {	AB,	AC,	BC,	A,	B,	C,	All	}
CostReduction =	6400	2200	2600	760	680	2320	-	

To materialized view All given S, we compute the benefit $B(All, S) = \sum_{w \leq All} B_w, where\ w = ViewToMat$

For each view w covered by All, we compute $C(AB) - C(All)$, because AB is dominated by All.

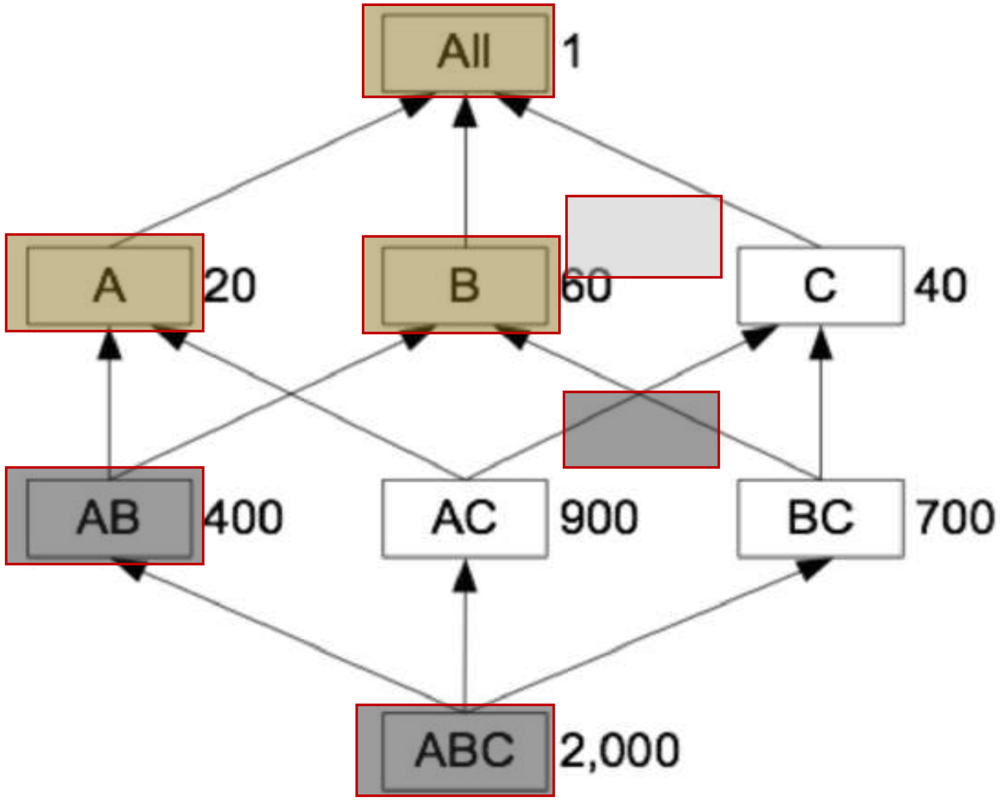
Hence, the cost reduction = $C(AB) - C(All) = 400 - 1.$

$B(All, S) = \sum_{w \leq All} B_w = 399$



S = {	**ABC,**	**AB**	**}**					
ViewToMat = w {	AB,	AC,	BC,	A,	B,	C,	All	}
CostReduction =	6400	2200	2600	760	680	2320	399	

The highest cost reduction from this iteration, 2600; that is by realizing the view BC. Hence, view BC will be materialized.



Third iteration

S = {	ABC,	AB,	BC	}				
ViewToMat = w {	AB,	AC,	BC,	A,	B,	C,	All	}
CostReduction =	6400	1100	2600	760	680	1020	399	

Following the same process, the cost reduction for views AC, A, B, C and All are computed to be 1100, 700, 680, 1020 and 399 respectively.

The highest cost reduction in this iteration is 1100 by realizing the view AC.

Since we only need to realized 3 additional views, the final materialized views created will be AB, AC and BC.