CS5226 Lecture 5
Materialized Views Tuning

Materialized View Tuning

```
Q:  select  R.b
    from    R, S
    where   R.a = S.x
    and     S.y > 100

V:  select  R.a, R.b, S.x, S.y
    from    R, S
    where   R.a = S.x

Q:  select  b
    from    V
    where   y > 100
```

Query answering using views

- Given a view V and a query Q, V subsumes Q if
  - V contains at least all the tuples required in Q, and
  - it is possible to filter out the irrevalent tuples from V
- If a view V subsumes a query Q, then V can be used to answer Q

$\mathcal{L}_{MV}$: MV definition language

select	$S_1, S_2, \cdots$	– projection columns
from	$T_1, T_2, \cdots$	– tables
where	J_1 **and** J_2 **and** $\cdots$	– equi-join predicates
	R_1 **and** R_2 **and** $\cdots$	– range predicates
	Z_1 **and** Z_2 **and** $\cdots$	– residual predicates
group by	$G_1, G_2, \cdots$	– grouping columns

- Each range predicate is a disjunction of open/closed intervals over same column
- Example:
  $R_1 = (a \in [10, 30]) \vee (a \in [50, 70]) \vee (a \geq 100)$

$\mathcal{L}_{MV}$: MV definition language (cont.)

Materialized view denoted by $V = (S, T, J, R, Z, G)$

- S = set of colums in the select clause
- T = set of tables
- J = set of join predicates
- R = set of range predicates
- Z = set of residual predicates
- G = set of grouping columns

Example

select R.f, S.w, **sum**(R.g)
from R, S
where R.a = S.x
and R.b = S.y
and (R.c $\in$ $[10, 50]$ **or** R.c $\in$ $[100, 200]$)
and (S.w $\in$ $[40, 80]$ **or** S.w $\geq$ 200)
and R.d + S.v $<$ 100
and R.e * S.u $>$ 200
group by R.f, S.w

Predicate containment

- Let p_1 & p_2 be two predicates on table T
- p_1 is contained in p_2 ($p_1 \sqsubseteq p_2$) if p_2 is more general than p_1
- If $p_1 \sqsubseteq p_2$, then $\sigma_{p_1}(T) \subseteq \sigma_{p_2}(T)$

Example:

- p_1: $(a \in [10, 40]) \vee (a \in [100, 130])$
- p_2: $(a \in [20, 30]) \vee (a \in [120, 125])$
- p_3: $a \in [10, 130]$
- $p_2 \sqsubseteq p_1 \sqsubseteq p_3$

View matching algorithm

- Let Q be a query & V be a view
- If V & Q are in $\mathcal{L}_{MV}$, there exists simple & sufficient conditions to check if a V subsumes Q
- V matches Q if these sufficient conditions are satisfied
- V matches Q $\implies$ V subsumes Q
- To check if V's predicates match Q's predicates:
    1. join predicates in Q $\supseteq$ join predicates in V
    2. residual predicates in Q $\supseteq$ residual predicate in V
    3. for each range predicate column C:
       Q's predicate on C $\sqsubseteq$ V's predicate on C

Materialized view transformations

- Merging: merges two views into a single view
- Reduction: reduces a view into another view involving fewer tables

View merging: $V_1 \oplus V_2$

- $V_1 = (S_1, T, J_1, R_1, Z_1, G_1)$
- $V_2 = (S_2, T, J_2, R_2, Z_2, G_2)$
- $V_M = V_1 \oplus V_2$ denote the merging of views V_1 & V_2 when the following properties hold:
    1. V_M belongs to $\mathcal{L}_{MV}$
    2. Both V_1 & V_2 can be derived from V_M
    3. If the view matching algorithm matches V_1 or V_2 for a query Q, it also matches V_M for Q
    4. V_M cannot be further restricted with additional predicates such that it satisfies the above properties

Case 1: No grouping columns

- $V_1 = (S_1, T, J_1, R_1, Z_1, \emptyset)$
- $V_2 = (S_2, T, J_2, R_2, Z_2, \emptyset)$

$V_1 \oplus V_2$:	**select**	$S_1 \cup S_2$
	from	T
	where	(J_1 **and** R_1 **and** Z_1)
	or	(J_2 **and** R_2 **and** Z_2)

Case 1: No grouping columns (cont.)

$$\begin{aligned} & (J_1 \wedge R_1 \wedge Z_1) \vee (J_2 \wedge R_2 \wedge Z_2) \\ = \; & (J_1 \vee J_1) \wedge (R_1 \vee R_2) \wedge (Z_1 \vee Z_2) \wedge C \\ \sqsubseteq \; & (J_1 \vee J_1) \wedge (R_1 \vee R_2) \wedge (Z_1 \vee Z_2) \end{aligned}$$

where $C = (J_1 \vee R_2) \wedge (R_1 \vee Z_2) \wedge \cdots$

Case 1: No grouping columns (cont.)

$$
\begin{aligned}
J_1 &= J_1^1 \wedge J_1^2 \wedge J_1^3 \wedge \cdots \\
J_2 &= J_2^1 \wedge J_2^2 \wedge J_2^3 \wedge \cdots \\
J_1 \vee J_2 &= (J_1^1 \wedge J_1^2 \wedge J_1^3 \wedge \cdots) \vee (J_2^1 \wedge J_2^2 \wedge J_2^3 \wedge \cdots) \\
&= \bigwedge_{i,j} (J_1^i \vee J_2^j) \sqsubseteq \bigwedge_{J^k \in J_i \cap J_j} J^k
\end{aligned}
$$

Define $J_1 \cap J_2 = \bigwedge_{J^k \in J_i \cap J_j} J^k$

Example: $J_1 = $ (R.x = S.x) $\wedge$ (R.y = S.y)
$J_2 = $ (R.x = S.x) $\wedge$ (R.y = S.y) $\wedge$ (R.z = S.z)
$J_1 \cap J_2 = $ (R.x = S.x) $\wedge$ (R.y = S.y)

Case 1: No grouping columns (cont.)

$$\begin{aligned} Z_1 &= Z_1^1 \wedge Z_1^2 \wedge Z_1^3 \wedge \cdots \\ Z_2 &= Z_2^1 \wedge Z_2^2 \wedge Z_2^3 \wedge \cdots \\ Z_1 \vee Z_2 &\sqsubseteq \bigwedge_{Z^k \in Z_i \cap Z_j} Z^k \end{aligned}$$

Define $Z_1 \cap Z_2 = \bigwedge_{Z^k \in Z_i \cap Z_j} Z^k$

Example:

$Z_1 = (\text{R.x + S.x} < 10) \wedge (\text{R.y * S.y} \geq 100)$

$Z_2 = (\text{R.x + S.x} < 10) \wedge (\text{R.y} - \text{S.y} \leq 20)$

$Z_1 \cap Z_2 = (\text{R.x + S.x} < 10)$

Case 1: No grouping columns (cont.)

$$
\begin{aligned}
R_1 &= R_1^1 \wedge R_1^2 \wedge \cdots \\
R_2 &= R_2^1 \wedge R_2^2 \wedge \cdots \\
R_1 \vee R_2 &= (R_1^1 \wedge R_1^2 \wedge \cdots) \vee (R_2^1 \wedge R_2^2 \wedge \cdots) \\
&= \bigwedge_{i,j} (R_1^i \vee R_2^j)
\end{aligned}
$$

- Define $R_1 \sqcup R_2 = \bigwedge_{i,j}(R_1^i \vee R_2^j)$ where R_1^i & R_2^j defined over the same columns
- $R_1 \vee R_2 \sqsubseteq R_1 \sqcup R_2$

Case 1: No grouping columns (cont.)

Example 1:

$$
\begin{array}{rccc}
R_1 = & (a \in [10,20] \vee a \in [30,40]) & \wedge & b \in [20,30] \\
R_2 = & a \in [15,35] & \wedge & b \in [10,25] \\
\hline
R_1 \sqcup R_2 = & a \in [10,40] & \wedge & b \in [10,30]
\end{array}
$$

Example 2:

$$
\begin{array}{rccccc}
R_1 = & c \leq 40 & \wedge & d \in [10,30] & & \\
R_2 = & c \geq 30 & \wedge & d \in [40,80] & \wedge & e \leq 50 \\
\hline
R_1 \sqcup R_2 = & & & (d \in [10,30]) \vee (d \in [40,80]) & &
\end{array}
$$

Case 1: No grouping columns (cont.)

V_1: **select** a
from R
where $10 < c < 20$

V_2: **select** b
from R
where $15 < c < 30$

$S_1 \cup S_2$

V_M: **select** a, b
from R
where $10 < c < 30$

Case 1: No grouping columns (cont.)

V_1: **select** a
from R
where $10 < c < 20$

V_2: **select** b
from R
where $15 < c < 30$

V'_M: **select** a, b, c $\longleftarrow S_1 \cup S_2 \cup S'$
from R
where $10 < c < 30$

S' = additional required columns

- Columns in join/residual predicates in V_1/V_2 that are missing from $V_1 \oplus V_2$
- Columns in range predicates in V_1/V_2 that are missing from or modified in $V_1 \oplus V_2$

Case 1: No grouping columns (cont.)

- $V_1 = (S_1, T, J_1, R_1, Z_1, \emptyset)$
- $V_2 = (S_2, T, J_2, R_2, Z_2, \emptyset)$
- $V_1 \oplus V_2 = (S_1 \cup S_2 \cup S', T, J_1 \cap J_2, R_1 \sqcup R_2, Z_1 \cap Z_2, \emptyset)$
    - S' = additional required columns
    - $J_1 \cap J_2 = \bigwedge_{J^k \in J_i \cap J_j} J^k$
    - $Z_1 \cap Z_2 = \bigwedge_{Z^k \in Z_i \cap Z_j} Z^k$
    - $R_1 \sqcup R_2$ = $\bigwedge_{i,j} (R_1^i \vee R_2^j)$ where R_1^i & R_2^j defined over the same columns

Example

V_1: **select** x, y
from R, S
where R.x = S.y
and $10 < R.a < 20$
and $R.b < 10$
and $R.x + S.d < 8$

V_2: **select** y, z
from R, S
where R.x = S.y
and $15 < R.a < 50$
and $R.b > 5$
and $R.c > 5$
and $R.x + S.d < 8$
and $R.e * R.e = 2$

$V_1 \oplus V_2$: **select** x, y, z, a, b, c, e
from R, S
where R.x = S.y
and $10 < R.a < 50$
and $R.x + S.d < 8$

Case 2: Grouping columns

- $V_1 = (S_1, T, J_1, R_1, Z_1, G_1)$
- $V_2 = (S_2, T, J_2, R_2, Z_2, G_2)$
- $V_1 \oplus \ V_2 = (S_M, T, J_1 \cap J_2, R_1 \sqcup R_2, Z_1 \cap Z_2, G_M)$
- S_M consists of
  - set of columns obtained in case 1
  - the group-by columns in G_M that are not in V_1/V_2
  - If $G_M = \emptyset$, all the aggregates are unfolded into base-table columns
- If ($G_1 = \emptyset$) or ($G_2 = \emptyset$)
  - $G_M = \emptyset$
- If ($G_1 \neq \emptyset$) and ($G_2 \neq \emptyset$)
  - $G_M = G_1 \cup \ G_2 \cup$ (columns added to S_M)

Example 1

V_1: **select** R.x, **sum**(S.y)
from R, S
where R.x = S.y
and $10 < R.a < 20$
group by R.x

V_2: **select** R.x, R.z
from R, S
where R.x = S.y
and $15 < R.a < 50$

base-table column from unfolded aggregate

$V_1 \oplus V_2$: **select** $\overbrace{R.x, R.z, R.a,}^{\text{columns obtained from case 1}}$ S.y
from R, S
where R.x = S.y
and $10 < R.a < 50$

Example 2

V_1: **select** R.x, **sum**(S.y)
from R, S
where R.x = S.y
and $10 < R.a < 20$
group by R.x

V_3: **select** S.y, **sum**(S.z)
from R, S
where R.x = S.y
and $10 < R.a < 25$
group by S.y

$V_1 \oplus V_3$: **select** R.x, S.y, R.a, **sum**(S.y), **sum**(S.z)
from R, S
where R.x = S.y
and $10 < R.a < 25$
group by R.x, S.y, R.a

View reduction

```
Q:  select  R.a, R.b, S.c
    from    R, S
    where   R.x = S.y
    and     R.a = 15

V:  select  R.a, R.b, S.c
    from    R, S
    where   R.x = S.y

Q:  select  a, b, c
    from    V
    where   a = 15
```

View reduction (cont.)

```
Q:   select  R.a, R.b, S.c
     from    R, S
     where   R.x = S.y
     and     R.a = 15

V':  select  R.a, R.b, R.x
     from    R
     where   R.a = 15

Q:   select  V'.a, V'.b, S.c
     from    V', S
     where   V'.x = S.y
```

View reduction: $\rho(V)$

- $V = (S, T, J, R, Z, G)$
- Given a view V and $T' \subseteq T$, $V_R = \rho(V, T')$ denote a reduction of V when the following properties hold:
  1. V_R belongs to $\mathcal{L}_{MV}$
  2. The set of tables in V_R = T'
  3. V can be derived from V_R
  4. If the view matching algorithm matches V for a query Q, it also matches V_R for a subquery of Q

View reduction: $\rho(V)$ (cont.)

- $V = (S, T, J, R, Z, G)$
- $\rho(V, T') = (S', T', J', R', Z', G')$
- $J' \subseteq J$, $R' \subseteq R$, $Z' \subseteq Z$, where each base table column referenced in J', R', and Z' refers exclusively to tables in T'
- S' consists of
  - all the columns in S that belong to tables in T', and
  - all columns in T' referenced in $J - J'$, $R - R'$, and $Z - Z'$
- $G' = \emptyset$

Invalid view reductions

- A view reduction $\rho(V, T')$ is invalid if it contains some cartesian product and V does not contain any cartesian product

Example

V:	**select**	R.c, S.c
	from	R, S
	where	R.x = S.y
	and	R.a $\in [10, 50]$
	and	S.a $\in [20, 30]$
	and	R.b + S.b < 10
	group by	R.c, S.c
$\rho(V, \{R\})$:	**select**	R.c, R.b, R.x
	from	R
	where	R.a $\in [10, 50]$

Indexes over materialized views

Let V denote a view

Let V(C) denote an index with columns C defined over V

Unified merging operator

- Consider two indexes $I_1 = V_1(C_1)$ and $I_2 = V_2(C_2)$
- If the two views V_1 & V_2 are merged to $V_M = V_1 \oplus V_2$, we also merge their indexes to $I_M = I_1 \oplus I_2$

Unified merging operator

- Promoting an index $I_1 = V_1(C_1)$ to V_M (denoted by $I_1 \uparrow V_M$) creates an index I' over V_M that can be used whenever I_1 is used
- $I_1 \uparrow V_M = V_M(C_1, X)$, where X consists of
  - Columns in V_1's join/residual predicates that are missing from V_M
  - Columns in V_1's range predicates that are missing from or modified in V_M
- Merging indexes over views

$$I_1 \oplus I_2 = (I_1 \uparrow V_M) \oplus (I_2 \uparrow V_M)$$

Unified merging operator (cont.)

V_1: **select** x, y
from R
where a $\in [10, 20]$

$I_1 = V_1(x)$

V_2: **select** y, z
from R
where a $\in [15, 30]$

$I_2 = V_2(z)$

$V_M = V_1 \oplus V_2$: **select** a, x, y, z
from R
where a $\in [10, 30]$

- $(I_1 \uparrow V_M) = V_M(x, a)$
- $(I_2 \uparrow V_M) = V_M(z, a)$
- $I_1 \oplus I_2 = V_M(x, a) \oplus V_M(z, a) = V_M(x, a, z)$

References

Required Readings

- N. Bruno, s. Chaudhuri, Physical design refinment: the merge-reduce approach, TODS 32(4), 2007.

Additional Readings

- Chapter 8 of Bruno's book
- N. Bruno and S. Chaudhuri, Automatic Physical Database Tuning: A Relaxation-based Approach, SIGMOD 2005.