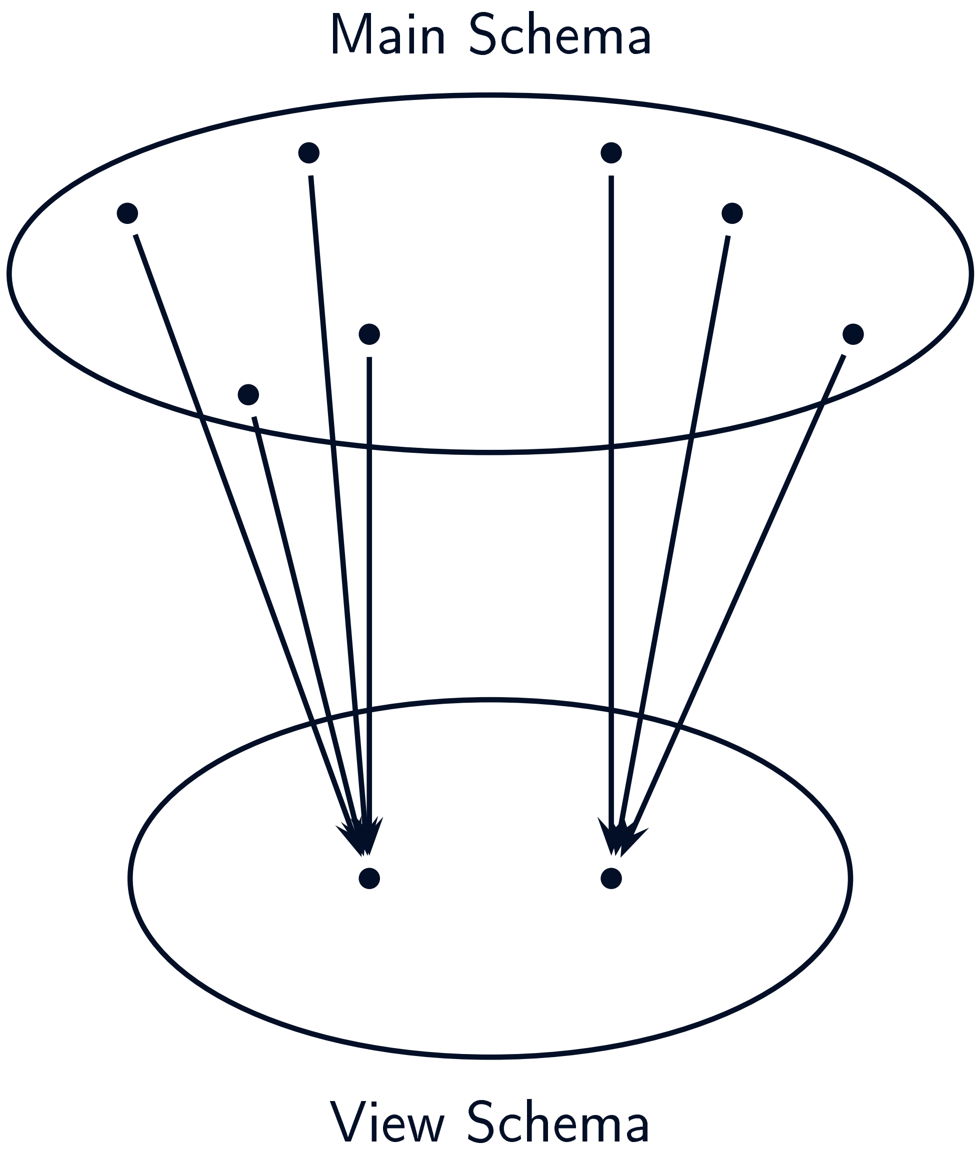
Optimal Complements for a Class of Relational Views

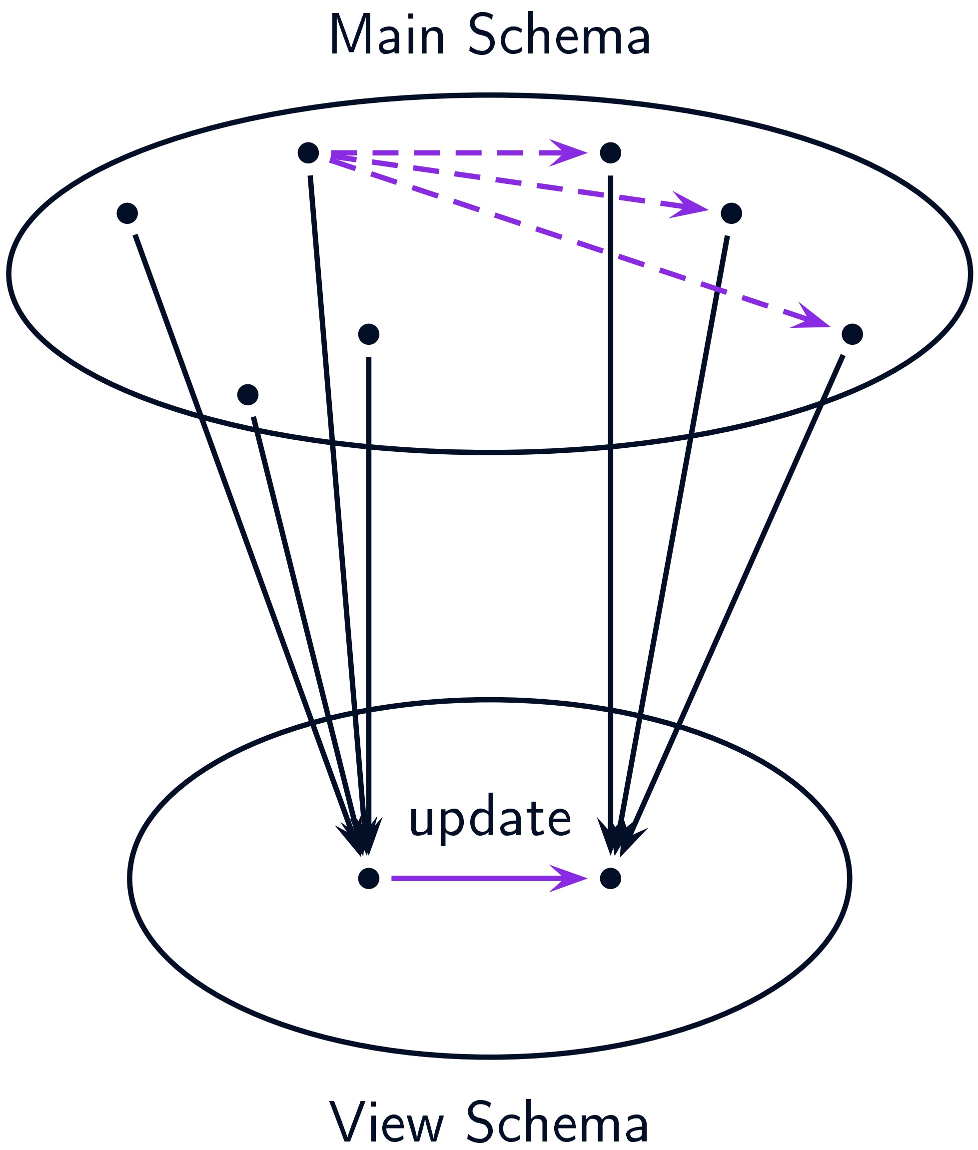
Stephen J. Hegner
Umeå University
Department of Computing Science
Sweden

- On the underlying states, the view mapping is generally *surjective* (onto) but not *injective* (one-to-one).



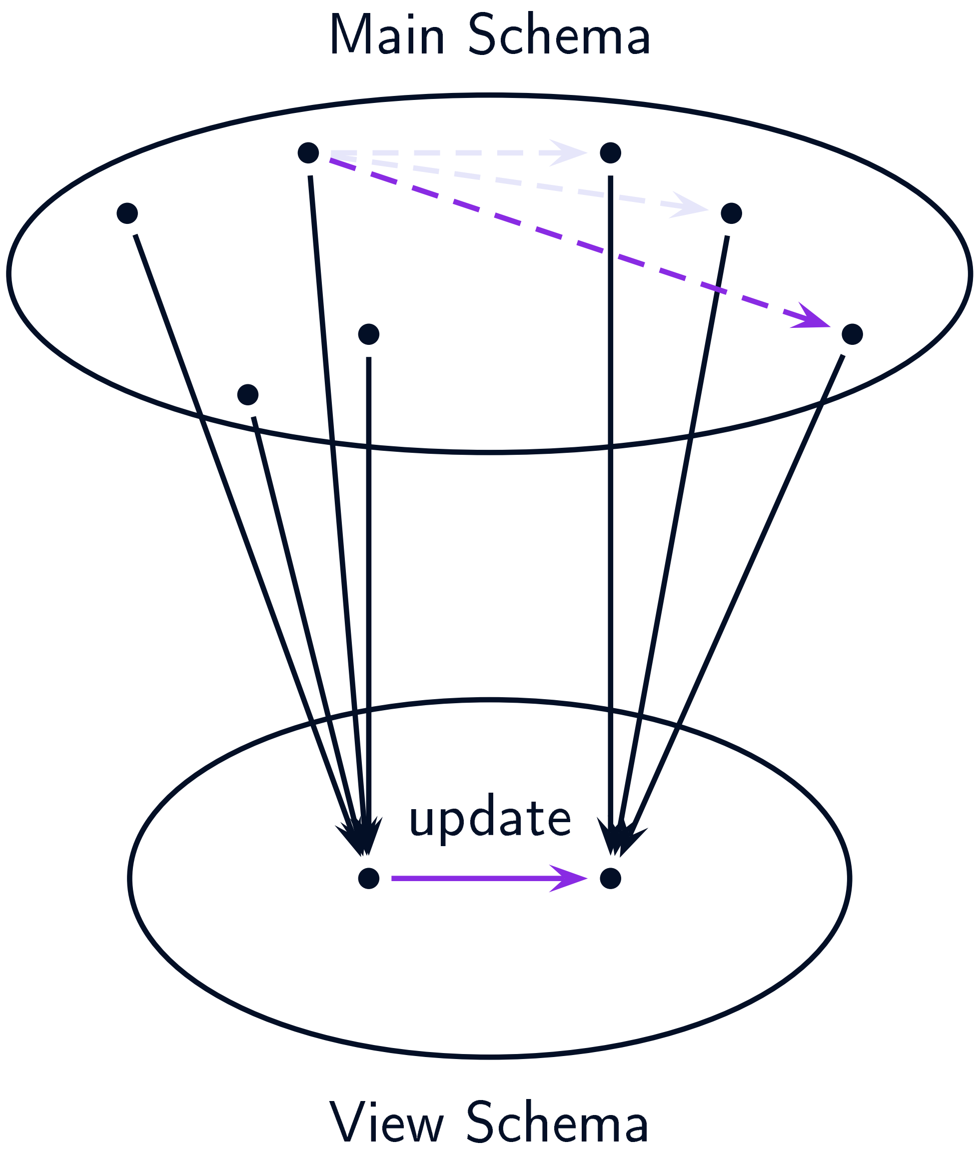
The Update Problem for Database Views

- On the underlying states, the view mapping is generally *surjective* (onto) but not *injective* (one-to-one).
- Thus, a view update has many possible *reflections* to the main schema.



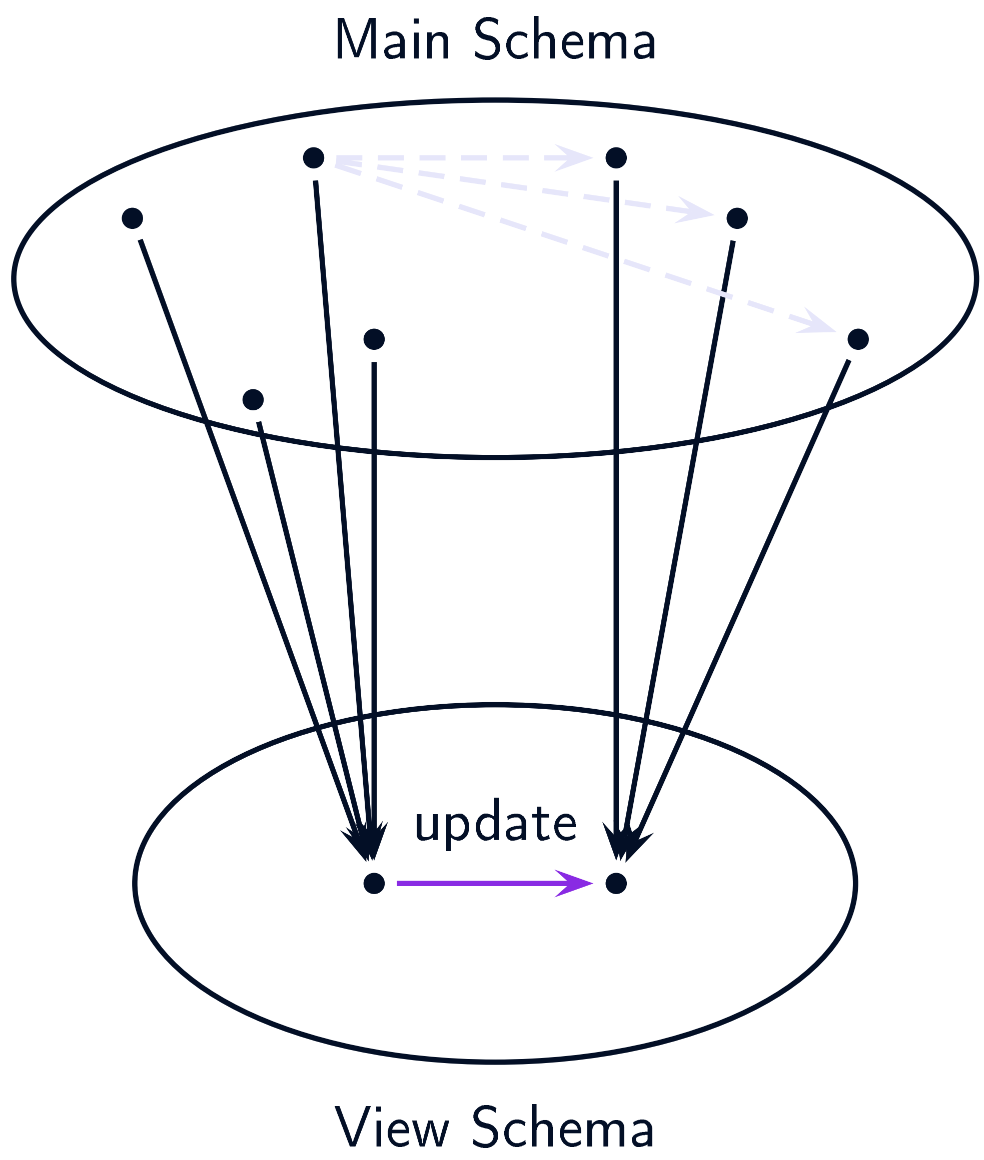
The Update Problem for Database Views

- On the underlying states, the view mapping is generally *surjective* (onto) but not *injective* (one-to-one).
- Thus, a view update has many possible *reflections* to the main schema.
- The problem of identifying a suitable reflection is known as the *update translation problem* or *update reflection problem*.

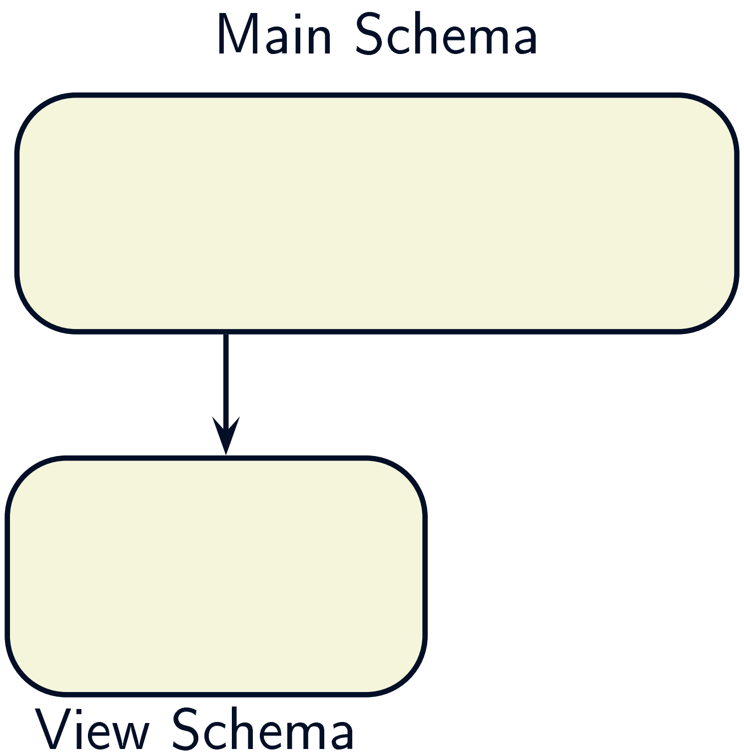


The Update Problem for Database Views

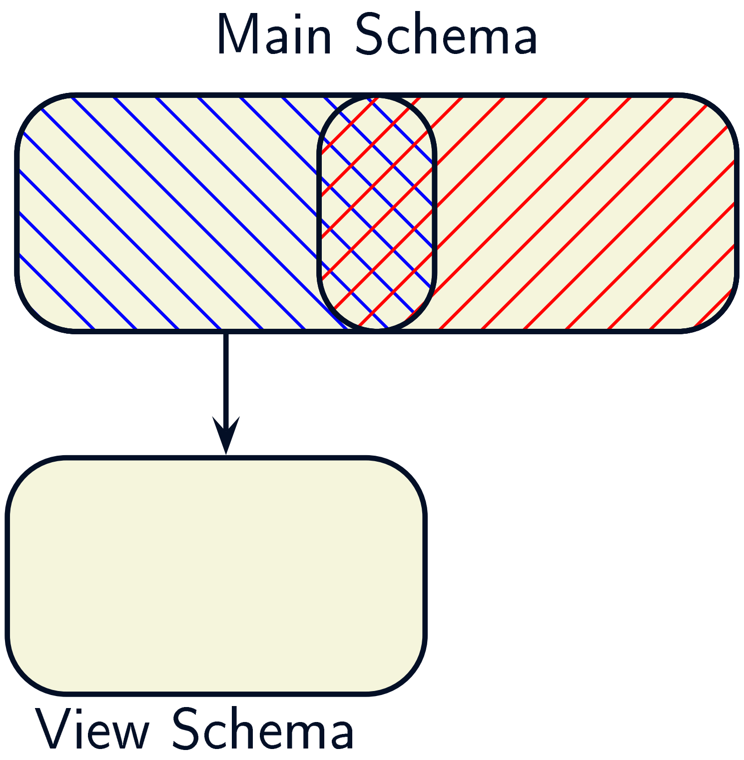
- On the underlying states, the view mapping is generally *surjective* (onto) but not *injective* (one-to-one).
- Thus, a view update has many possible *reflections* to the main schema.
- The problem of identifying a suitable reflection is known as the *update translation problem* or *update reflection problem*.
- With a reasonable definition of suitability, it may not be the case that every view update has a suitable translation.



The Gold Standard — the Constant-Complement Strategy

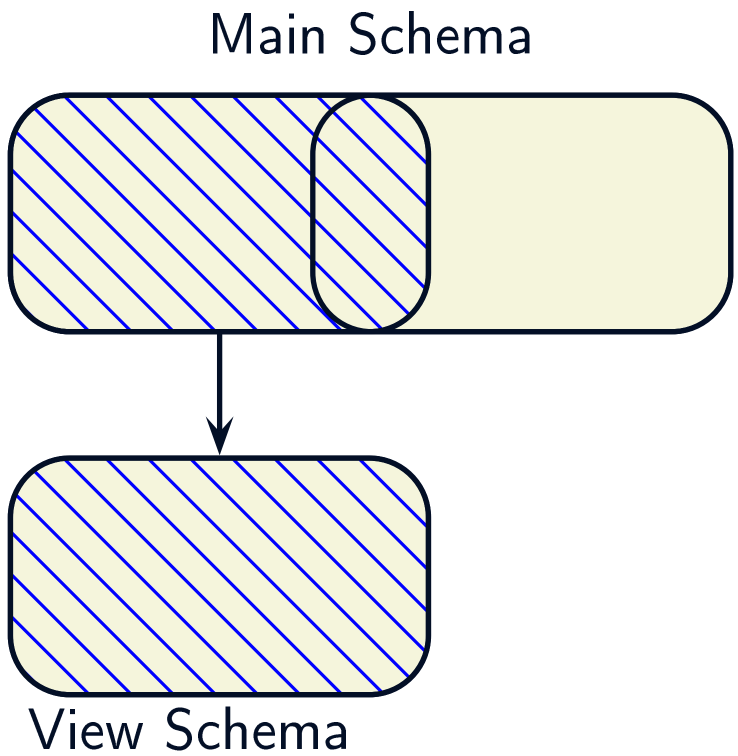


- In the constant-complement strategy [Bancilhon and Spyratos 81], [Hegner 04 AMAI], the main schema is decomposed into two *meet-complementary* views.



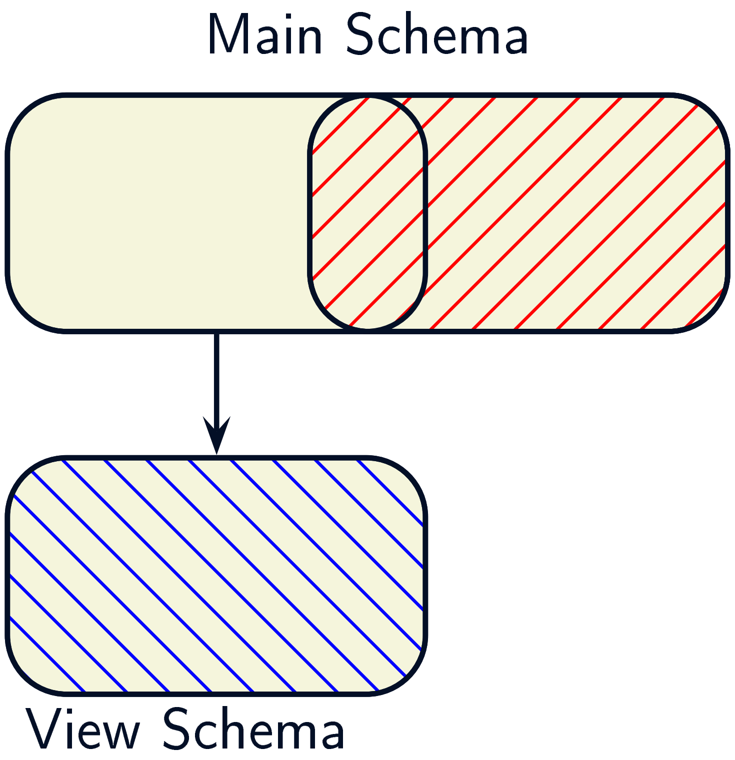
The Gold Standard — the Constant-Complement Strategy

- In the constant-complement strategy [Bancilhon and Spyratos 81], [Hegner 04 AMAI], the main schema is decomposed into two *meet-complementary* views.
- One is isomorphic to the view schema and tracks its updates exactly.



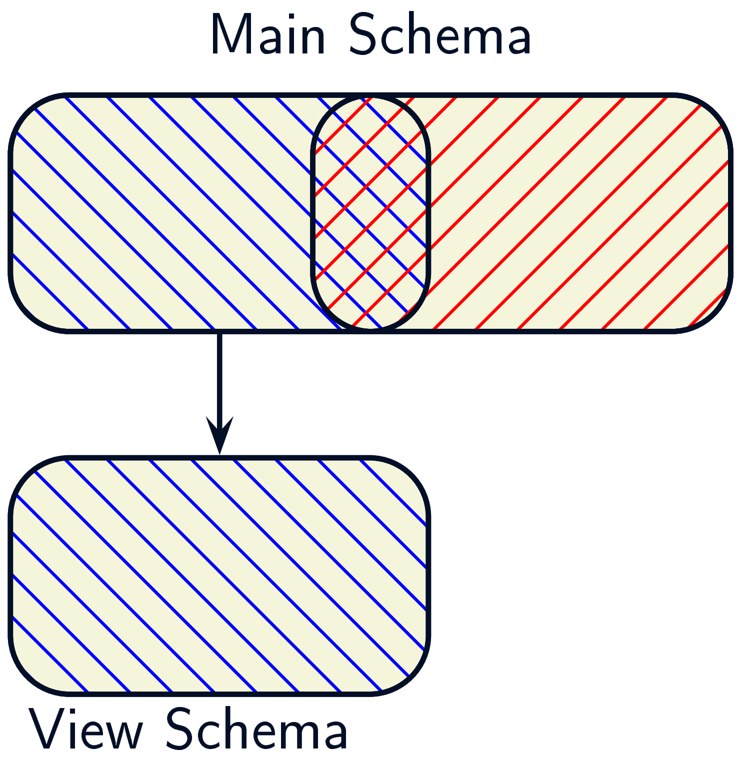
The Gold Standard — the Constant-Complement Strategy

- In the constant-complement strategy [Bancilhon and Spyratos 81], [Hegner 04 AMAI], the main schema is decomposed into two *meet-complementary* views.
- One is isomorphic to the view schema and tracks its updates exactly.
- The other is held constant for all updates to the view.



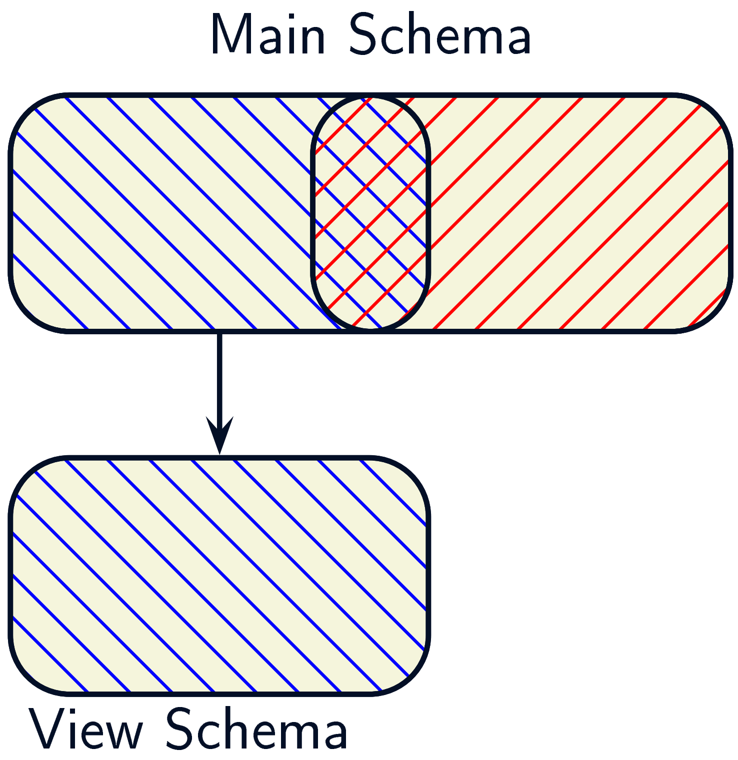
The Gold Standard — the Constant-Complement Strategy

- In the constant-complement strategy [Bancilhon and Spyratos 81], [Hegner 04 AMAI], the main schema is decomposed into two *meet-complementary* views.
- One is isomorphic to the view schema and tracks its updates exactly.
- The other is held constant for all updates to the view.
- Although it is somewhat limited in the view updates which it allows, they are supported in an optimal manner.



The Gold Standard — the Constant-Complement Strategy

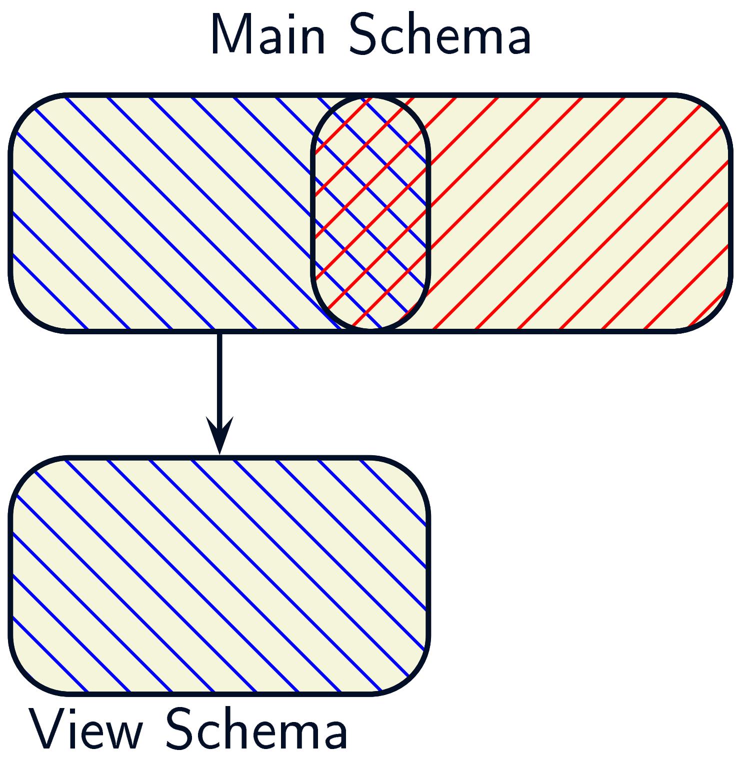
- In the constant-complement strategy [Bancilhon and Spyratos 81], [Hegner 04 AMAI], the main schema is decomposed into two *meet-complementary* views.
- One is isomorphic to the view schema and tracks its updates exactly.
- The other is held constant for all updates to the view.
- Although it is somewhat limited in the view updates which it allows, they are supported in an optimal manner.



- It can be shown [Hegner 03 AMAI] that this strategy is precisely that which avoids all *update anomalies*.

The Gold Standard — the Constant-Complement Strategy

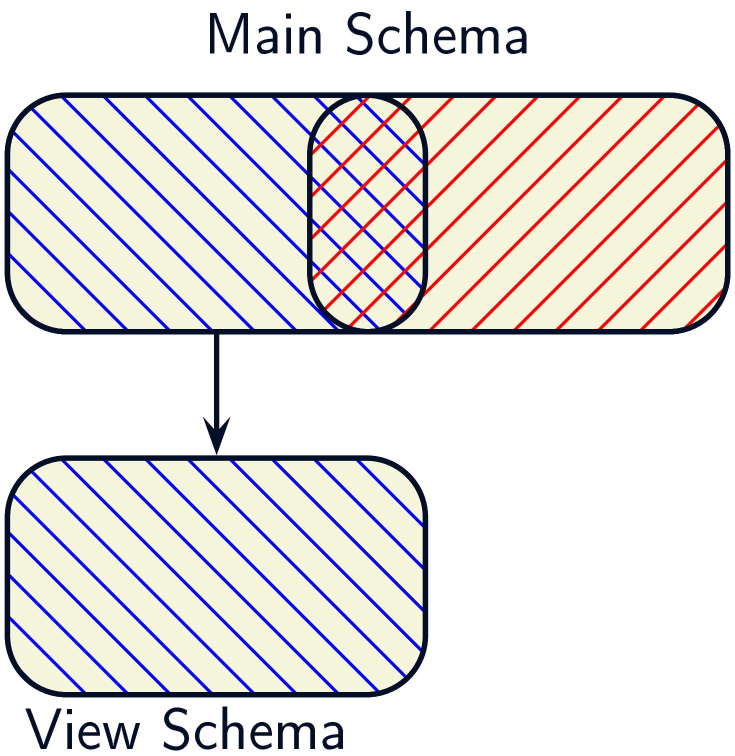
- In the constant-complement strategy [Bancilhon and Spyratos 81], [Hegner 04 AMAI], the main schema is decomposed into two *meet-complementary* views.
- One is isomorphic to the view schema and tracks its updates exactly.
- The other is held constant for all updates to the view.
- Although it is somewhat limited in the view updates which it allows, they are supported in an optimal manner.



- It can be shown [Hegner 03 AMAI] that this strategy is precisely that which avoids all *update anomalies*.
- However, this is complicated by the *complement uniqueness problem*.

The Gold Standard — the Constant-Complement Strategy

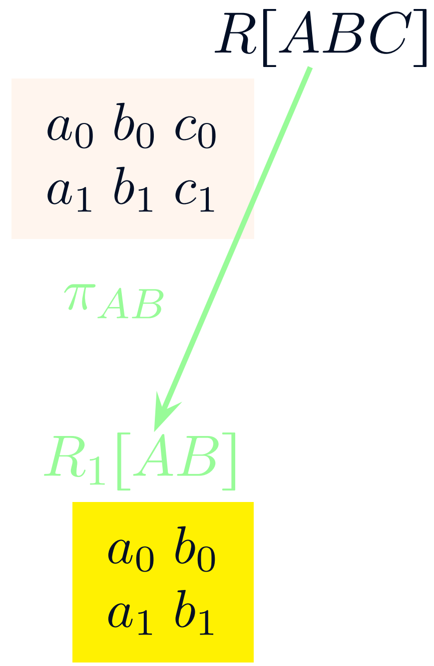
- In the constant-complement strategy [Bancilhon and Spyratos 81], [Hegner 04 AMAI], the main schema is decomposed into two *meet-complementary* views.
- One is isomorphic to the view schema and tracks its updates exactly.
- The other is held constant for all updates to the view.
- Although it is somewhat limited in the view updates which it allows, they are supported in an optimal manner.



- It can be shown [Hegner 03 AMAI] that this strategy is precisely that which avoids all *update anomalies*.
- However, this is complicated by the *complement uniqueness problem*.
- Some examples will help illustrate these ideas.

- Consider the classical example to the right.

Main Schema $\mathbf{E}_1$
Constraint: $\bowtie [AB, BC]$

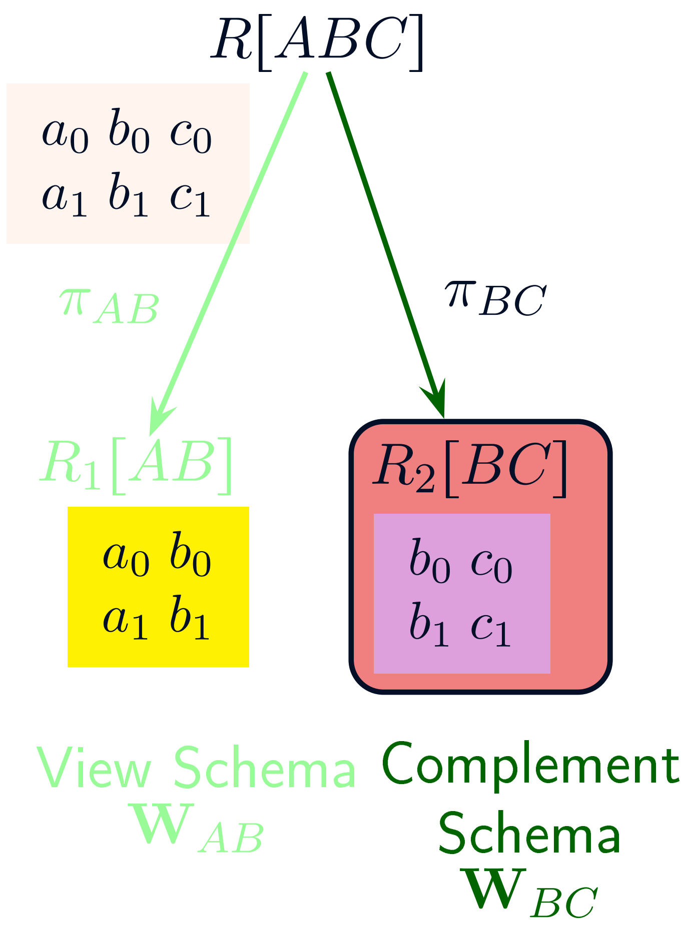


View Schema
$\mathbf{W}_{AB}$

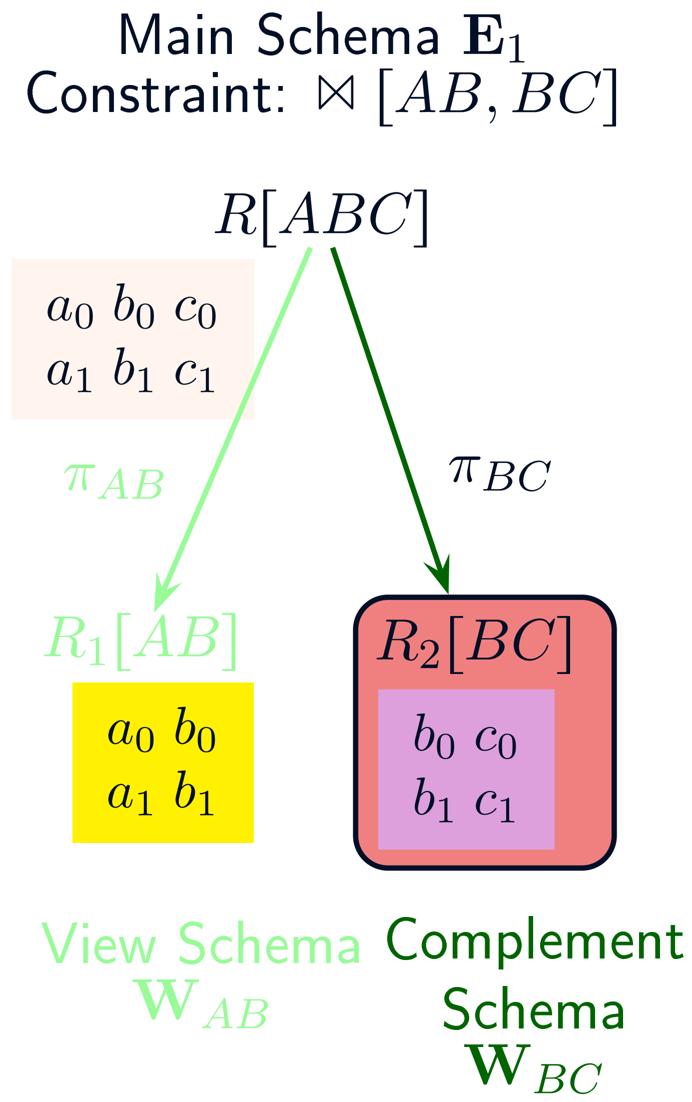
The Idea of Constant-Complement by Example

- Consider the classical example to the right.
- A natural complement to the AB-projection is the BC-projection.

Main Schema $\mathbf{E}_1$
Constraint: $\bowtie [AB, BC]$



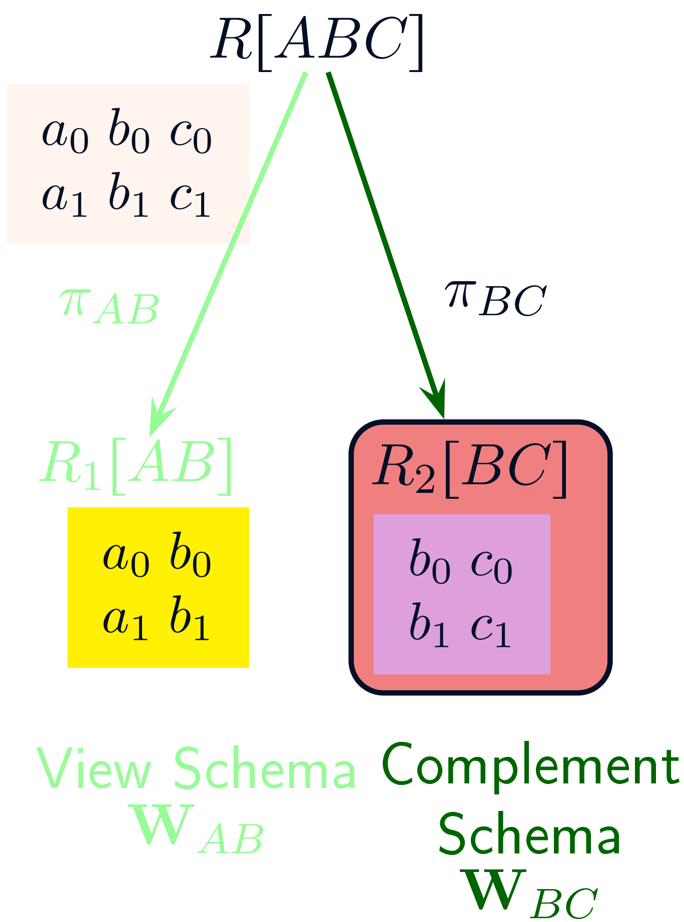
- Consider the classical example to the right.
- A natural complement to the AB-projection is the BC-projection.
- The *decomposed schema* $\mathbf{W}_{AB} \otimes \mathbf{W}_{BC}$ has relation symbols $R_[AB]$ and $R_2[BC]$; the legal database are all states which are join compatible on B.



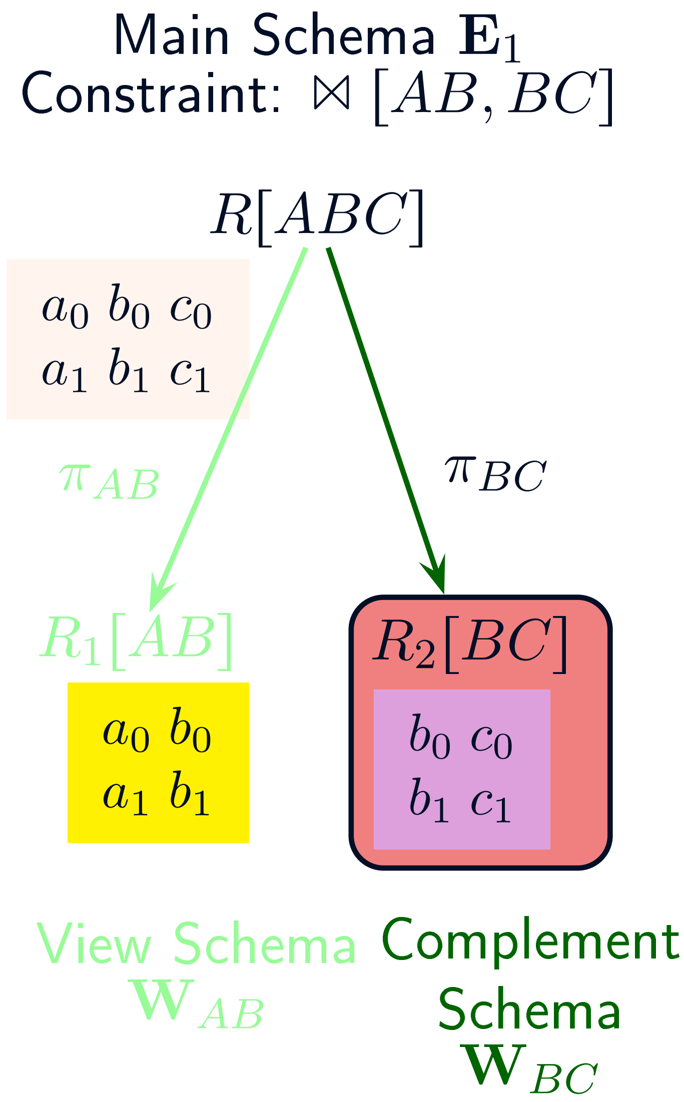
The Idea of Constant-Complement by Example

- Consider the classical example to the right.
- A natural complement to the AB-projection is the BC-projection.
- The *decomposed schema* $\mathbf{W}_{AB} \otimes \mathbf{W}_{BC}$ has relation symbols $R_{[}AB]$ and $R_2[BC]$; the legal database are all states which are join compatible on B.
- The *decomposition mapping* $\mathbf{E}_1 \to \mathbf{W}_{AB} \otimes \mathbf{W}_{BC}$, and is always bijective for complements.

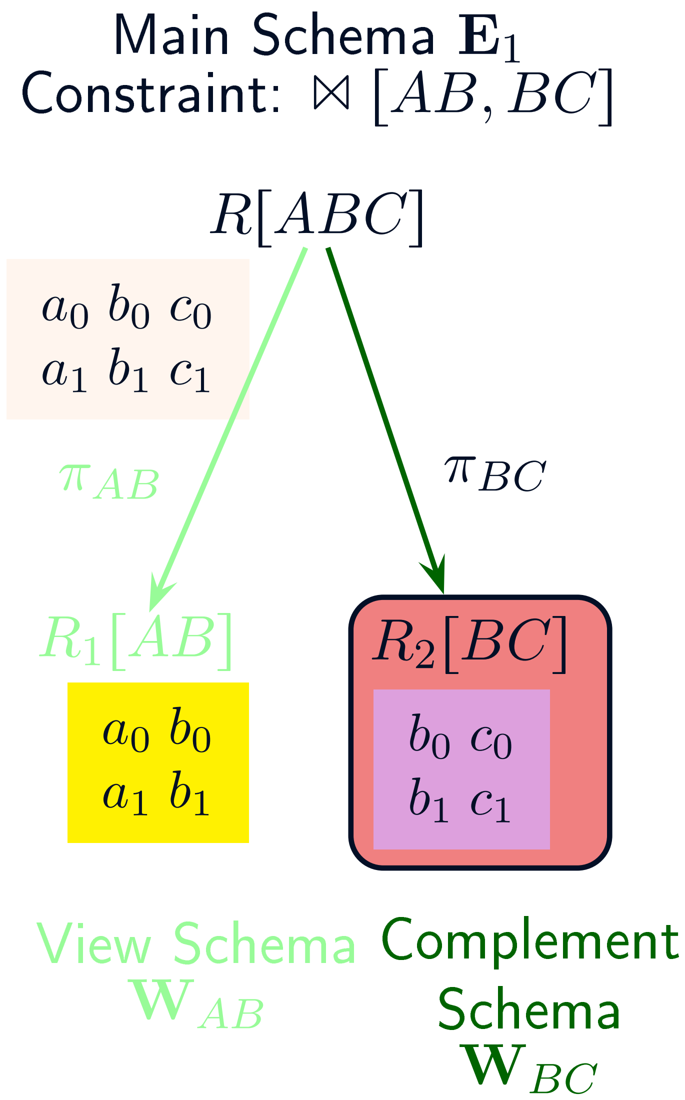
Main Schema $\mathbf{E}_1$
Constraint: $\bowtie [AB, BC]$



- Consider the classical example to the right.
- A natural complement to the AB-projection is the BC-projection.
- The *decomposed schema* $\mathbf{W}_{AB} \otimes \mathbf{W}_{BC}$ has relation symbols $R_{[}AB]$ and $R_2[BC]$; the legal database are all states which are join compatible on B.
- The *decomposition mapping* $\mathbf{E}_1 \to \mathbf{W}_{AB} \otimes \mathbf{W}_{BC}$, and is always bijective for complements.
- The *reconstruction mapping* $\mathbf{W}_{AB} \otimes \mathbf{W}_{BC} \to \mathbf{W}_1$ is the inverse of the decomposition mapping. It is the natural join in this case.



- Consider the classical example to the right.
- A natural complement to the AB-projection is the BC-projection.
- The *decomposed schema* $\mathbf{W}_{AB} \otimes \mathbf{W}_{BC}$ has relation symbols $R_{[}AB]$ and $R_2[BC]$; the legal database are all states which are join compatible on B.
- The *decomposition mapping* $\mathbf{E}_1 \to \mathbf{W}_{AB} \otimes \mathbf{W}_{BC}$, and is always bijective for complements.
- The *reconstruction mapping* $\mathbf{W}_{AB} \otimes \mathbf{W}_{BC} \to \mathbf{W}_1$ is the inverse of the decomposition mapping. It is the natural join in this case.
- The view which is the projection on B is the *meet* of $\mathbf{W}_{AB}$ and $\mathbf{W}_{BC}$, and is precisely that which must be held constant under a constant-complement update.



- Given is the following two-relation main schema.

Main Schema $\mathbf{E}_0$
No dependencies

$R[A]$ $S[A]$

The Problem of Complement Uniqueness

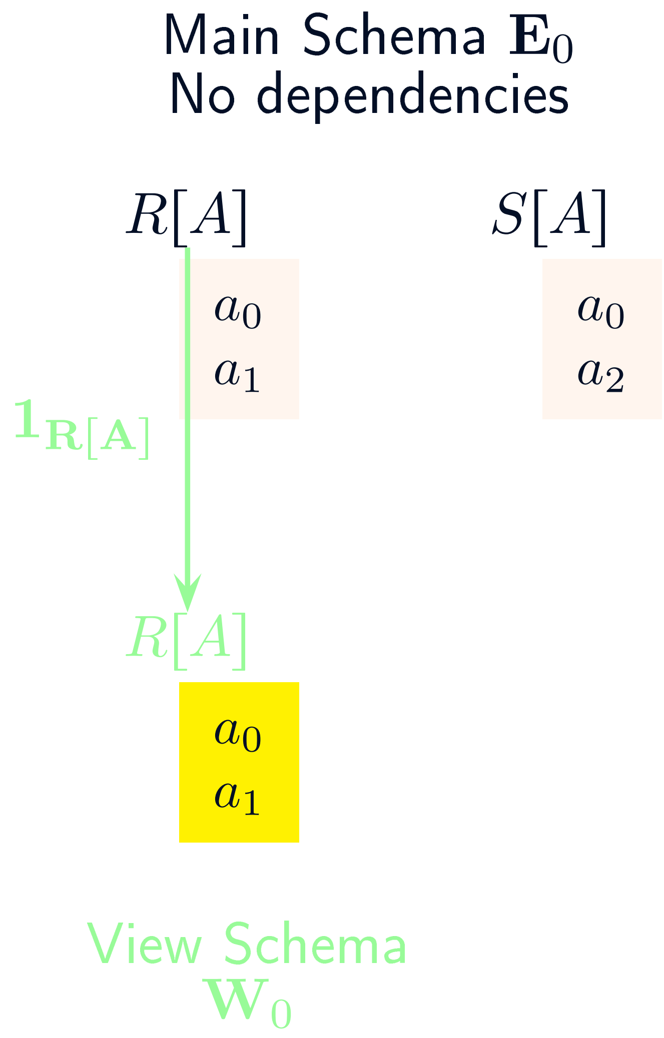
- Given is the following two-relation main schema.

Main Schema $\mathbf{E}_0$
No dependencies

$R[A]$
a_0
a_1

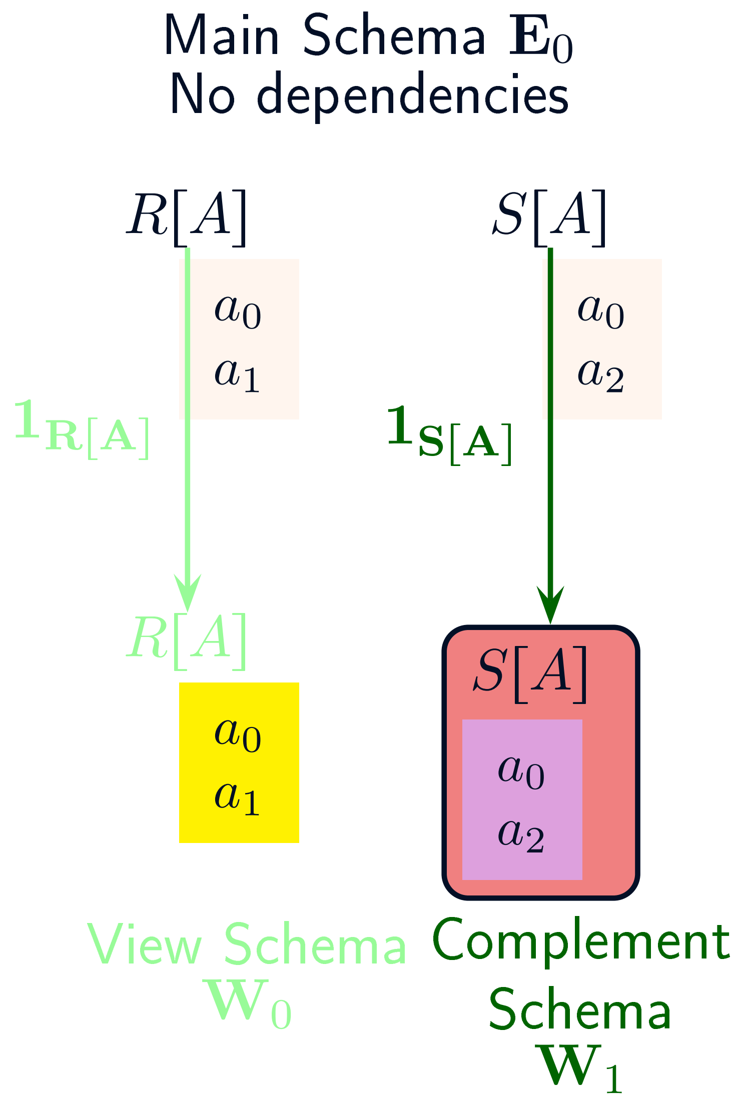
$S[A]$
a_0
a_2

- Given is the following two-relation main schema.
- The view schema $\mathbf{W}_0$ to be updated is that which preserves $R[A]$ but discards $S[A]$.



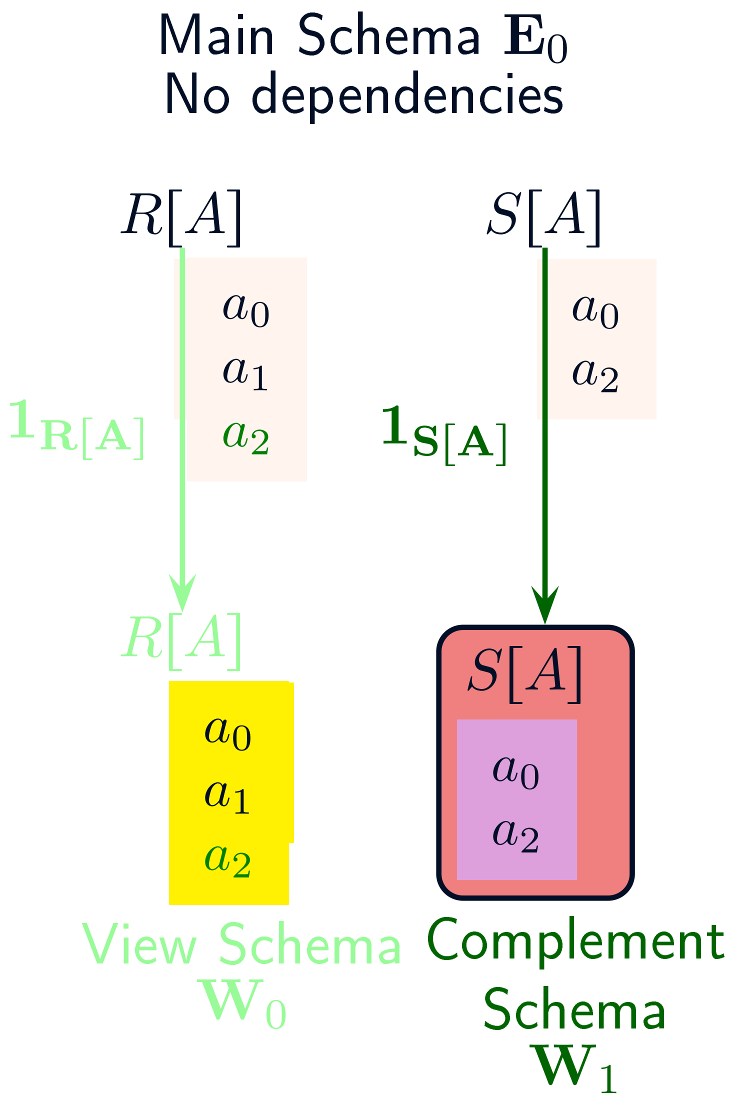
The Problem of Complement Uniqueness

- Given is the following two-relation main schema.
- The view schema $\mathbf{W}_0$ to be updated is that which preserves $R[A]$ but discards $S[A]$.
- The *natural complement* $\mathbf{W}_1$ is the schema which preserves $S[A]$ but discards $R[A]$.

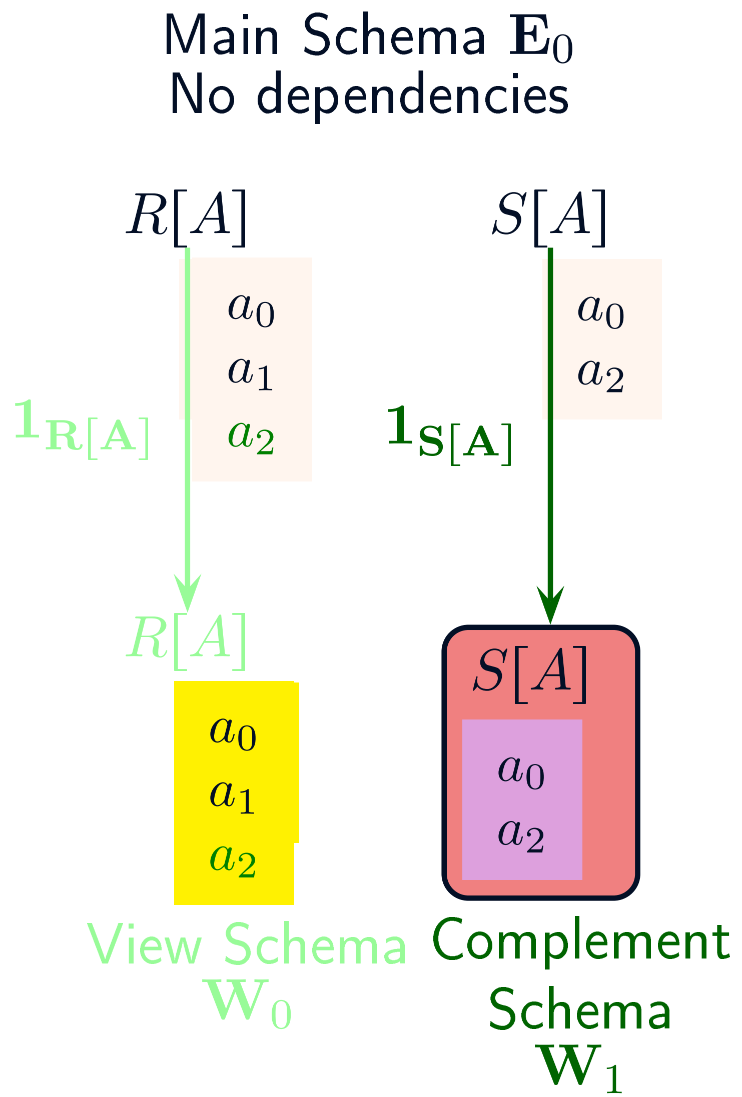


The Problem of Complement Uniqueness

- Given is the following two-relation main schema.
- The view schema $\mathbf{W}_0$ to be updated is that which preserves $R[A]$ but discards $S[A]$.
- The *natural complement* $\mathbf{W}_1$ is the schema which preserves $S[A]$ but discards $R[A]$.
- With $\mathbf{W}_1$ constant, all updates to $R[A]$ are allowed.

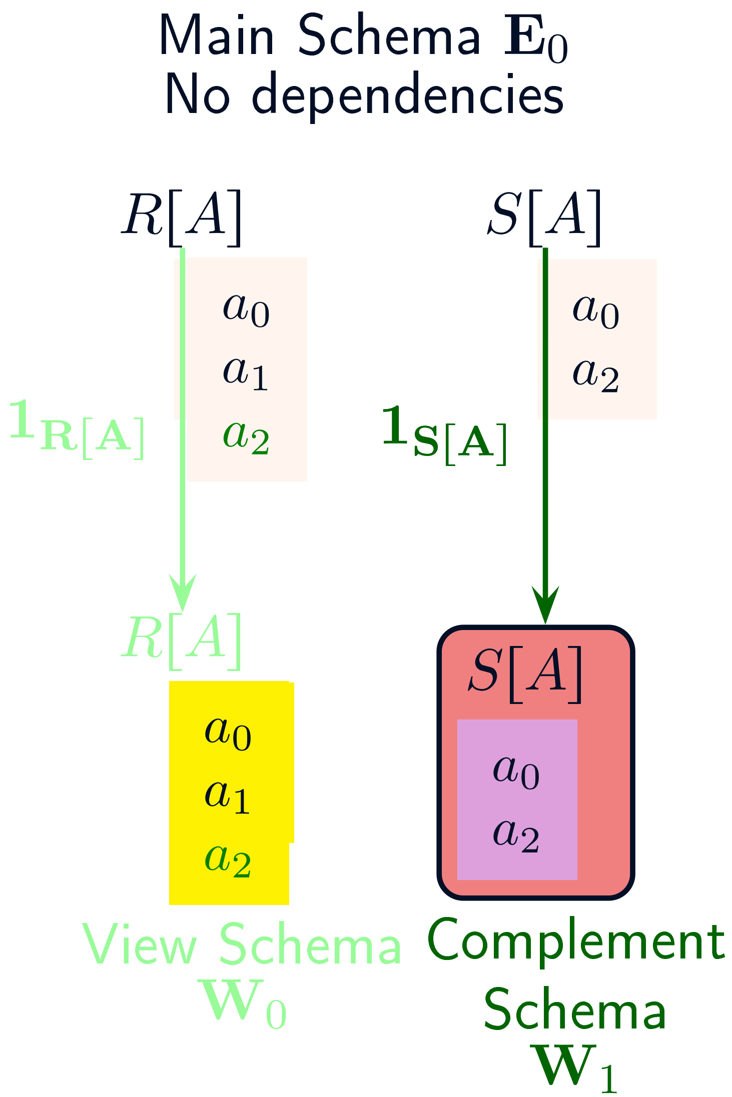


- Given is the following two-relation main schema.
- The view schema $\mathbf{W}_0$ to be updated is that which preserves $R[A]$ but discards $S[A]$.
- The *natural complement* $\mathbf{W}_1$ is the schema which preserves $S[A]$ but discards $R[A]$.
- With $\mathbf{W}_1$ constant, all updates to $R[A]$ are allowed.
- Clearly, this is the only reasonable update strategy for $\mathbf{W}_0$.

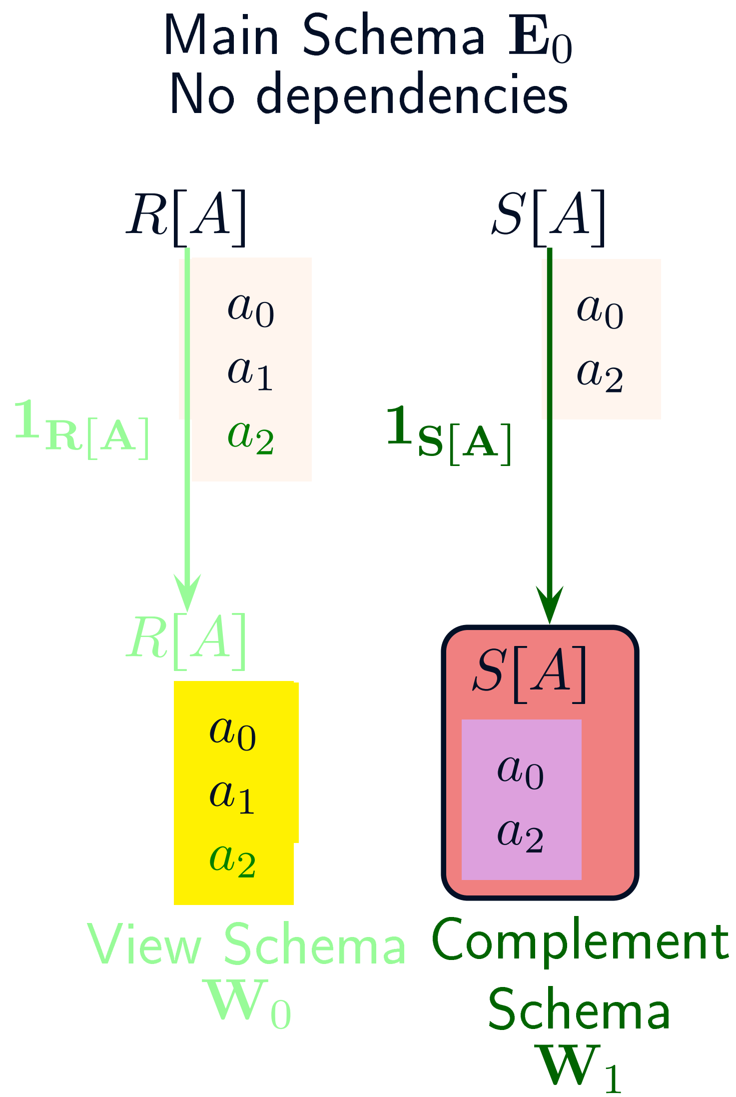


The Problem of Complement Uniqueness

- Given is the following two-relation main schema.
- The view schema $\mathbf{W}_0$ to be updated is that which preserves $R[A]$ but discards $S[A]$.
- The *natural complement* $\mathbf{W}_1$ is the schema which preserves $S[A]$ but discards $R[A]$.
- With $\mathbf{W}_1$ constant, all updates to $R[A]$ are allowed.
- Clearly, this is the only reasonable update strategy for $\mathbf{W}_0$.
- However, $\mathbf{W}_1$ does not define the only complement.



- Given is the following two-relation main schema.
- The view schema $\mathbf{W}_0$ to be updated is that which preserves $R[A]$ but discards $S[A]$.
- The *natural complement* $\mathbf{W}_1$ is the schema which preserves $S[A]$ but discards $R[A]$.
- With $\mathbf{W}_1$ constant, all updates to $R[A]$ are allowed.
- Clearly, this is the only reasonable update strategy for $\mathbf{W}_0$.
- However, $\mathbf{W}_1$ does not define the only complement.
- Without further restrictions, complements are almost never unique.



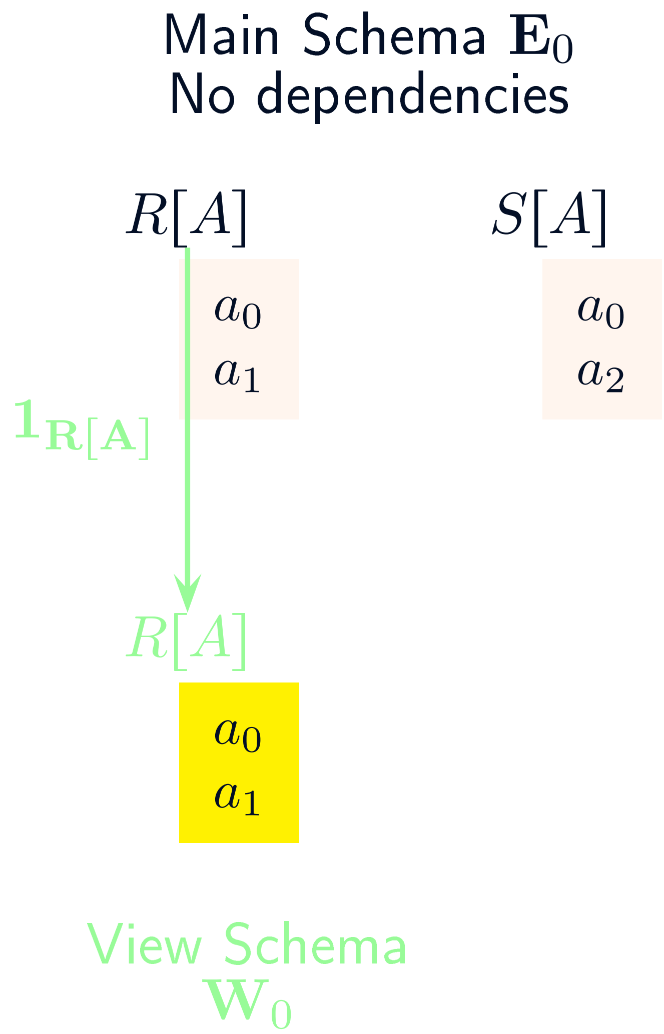
An Alternate Complement

- The main schema is unchanged.

Main Schema $\mathbf{E}_0$
No dependencies

$R[A]$ $S[A]$

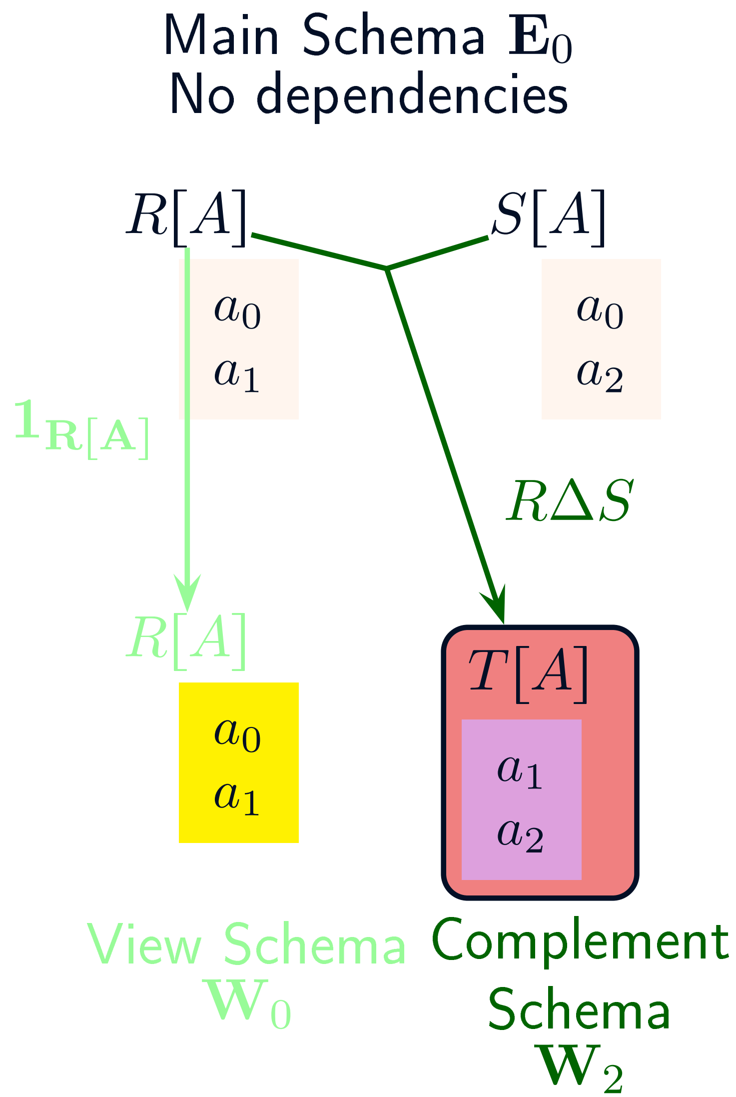
- The main schema is unchanged.
- The view schema $\mathbf{W}_0$ to be updated is also the same.



An Alternate Complement

- The main schema is unchanged.
- The view schema $\mathbf{W}_0$ to be updated is also the same.
- An alternative complement $\mathbf{W}_2$ is defined by the symmetric difference:

$$T[A] = (R[A] \backslash S[A]) \cup (S[A] \backslash R[A])$$

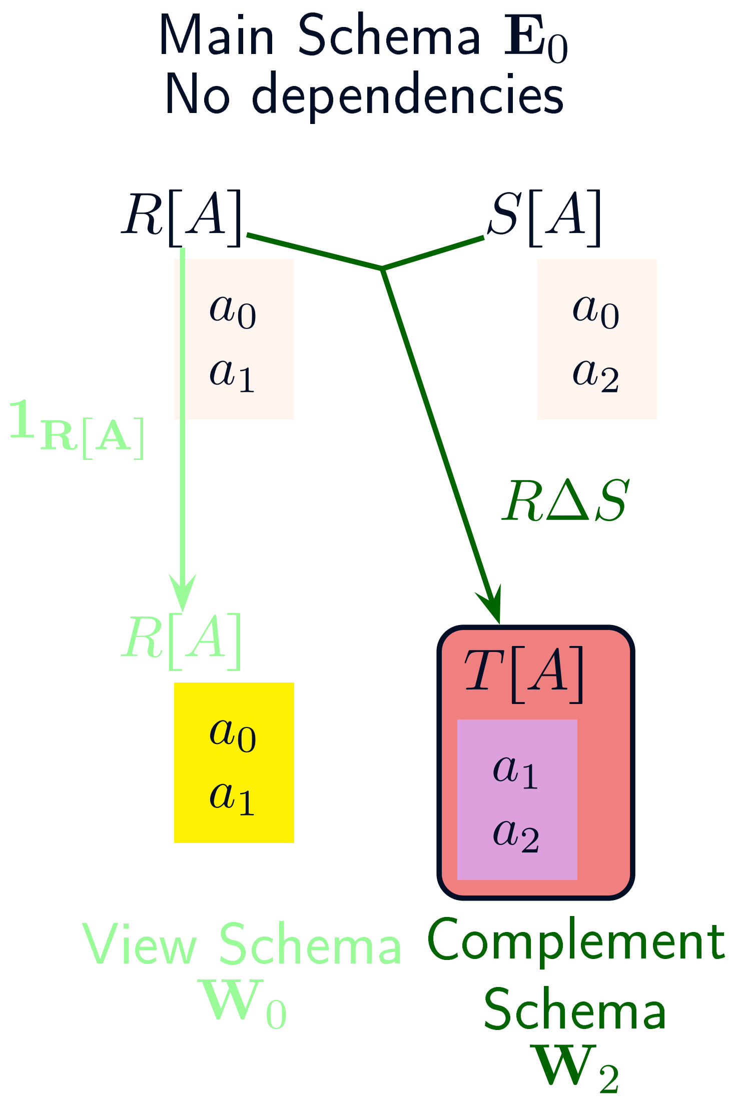


An Alternate Complement

- The main schema is unchanged.
- The view schema $\mathbf{W}_0$ to be updated is also the same.
- An alternative complement $\mathbf{W}_2$ is defined by the symmetric difference:

$$T[A] = (R[A] \backslash S[A]) \cup (S[A] \backslash R[A])$$

- With this alternative complement, the update strategy is different — $S[A]$ is altered.

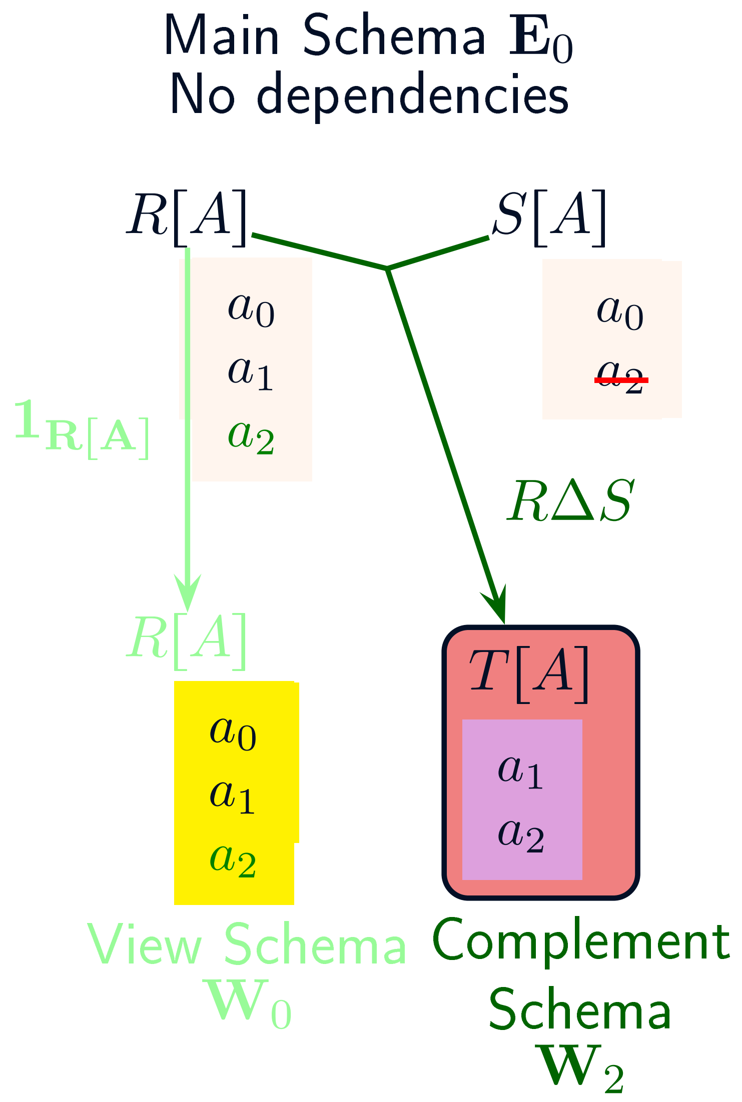


An Alternate Complement

- The main schema is unchanged.
- The view schema $\mathbf{W}_0$ to be updated is also the same.
- An alternative complement $\mathbf{W}_2$ is defined by the symmetric difference:

$$T[A] = (R[A] \backslash S[A]) \cup (S[A] \backslash R[A])$$

- With this alternative complement, the update strategy is different — $S[A]$ is altered.
- Clearly, this is not a desirable complement.

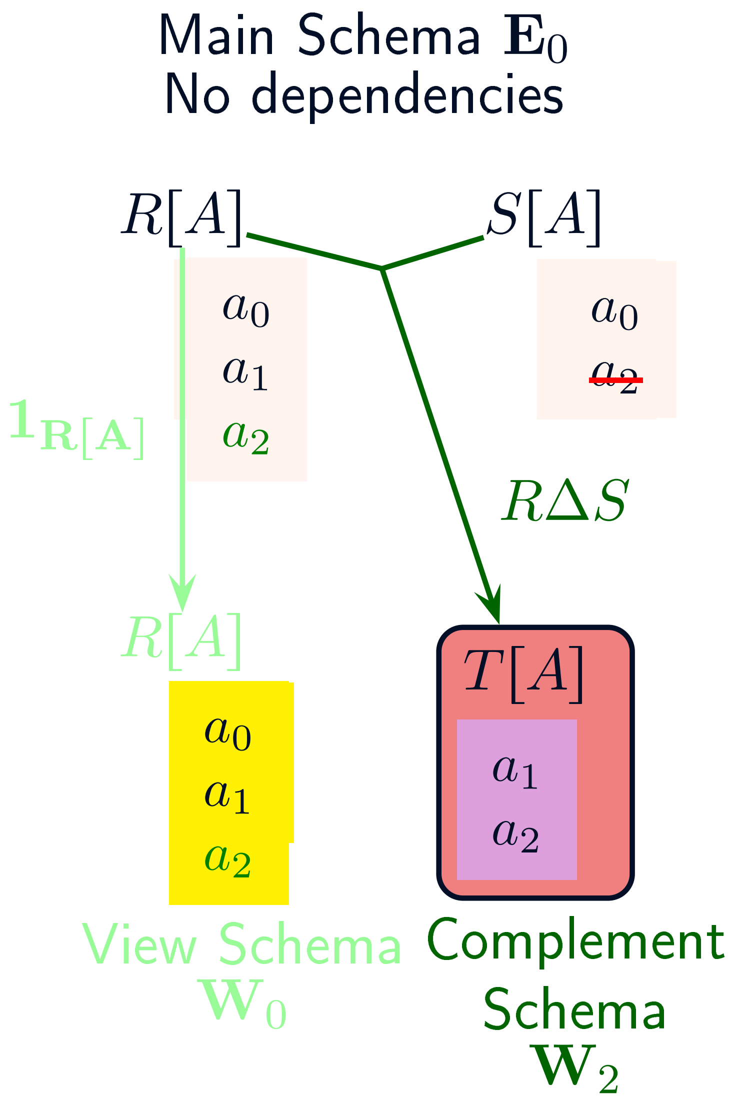


- The main schema is unchanged.
- The view schema $\mathbf{W}_0$ to be updated is also the same.
- An alternative complement $\mathbf{W}_2$ is defined by the symmetric difference:

$$T[A] = (R[A] \backslash S[A]) \cup (S[A] \backslash R[A])$$

- With this alternative complement, the update strategy is different — $S[A]$ is altered.
- Clearly, this is not a desirable complement.

Question: How can these two complements be distinguished formally?



The Sufficiency of Monotonicity

- Note that the symmetric difference mapping is not monotonic with respect to the natural order of database states.

The Sufficiency of Monotonicity

- Note that the symmetric difference mapping is not monotonic with respect to the natural order of database states.

Theorem: If the view to be updated and is defined by a monotonic morphism, then the reflection of a given view update to the main schema is independent of the choice of complement, provided that the complement is also defined by a monotonic morphism.

The Sufficiency of Monotonicity

- Note that the symmetric difference mapping is not monotonic with respect to the natural order of database states.

Theorem: If the view to be updated and is defined by a monotonic morphism, then the reflection of a given view update to the main schema is independent of the choice of complement, provided that the complement is also defined by a monotonic morphism.

Proof: [Hegner 04 AMAI], [Hegner 08 SDKB], [Hegner 09 LID], [Hegner 10 JUCS] □

The Sufficiency of Monotonicity

- Note that the symmetric difference mapping is not monotonic with respect to the natural order of database states.

Theorem: If the view to be updated and is defined by a monotonic morphism, then the reflection of a given view update to the main schema is independent of the choice of complement, provided that the complement is also defined by a monotonic morphism.

Proof: [Hegner 04 AMAI], [Hegner 08 SDKB], [Hegner 09 LID], [Hegner 10 JUCS] □

However: It is not necessarily the case that all such view updates may be realized using the same complement.

The Sufficiency of Monotonicity

- Note that the symmetric difference mapping is not monotonic with respect to the natural order of database states.

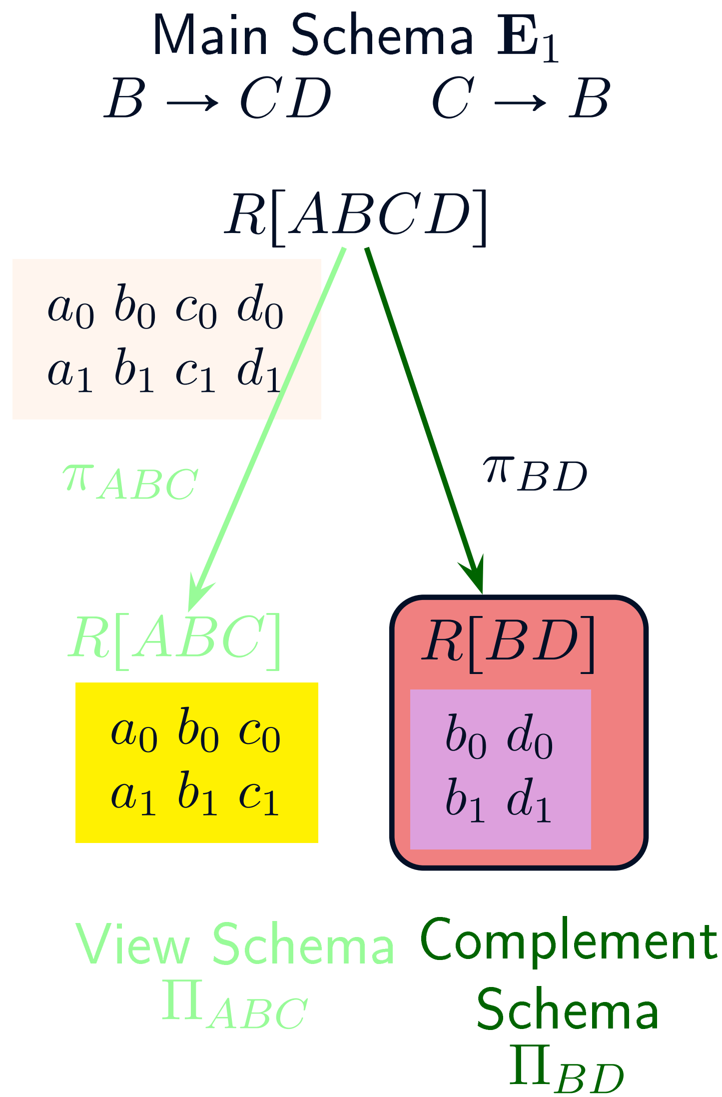
Theorem: If the view to be updated and is defined by a monotonic morphism, then the reflection of a given view update to the main schema is independent of the choice of complement, provided that the complement is also defined by a monotonic morphism.

Proof: [Hegner 04 AMAI], [Hegner 08 SDKB], [Hegner 09 LID], [Hegner 10 JUCS] □

However: It is not necessarily the case that all such view updates may be realized using the same complement.

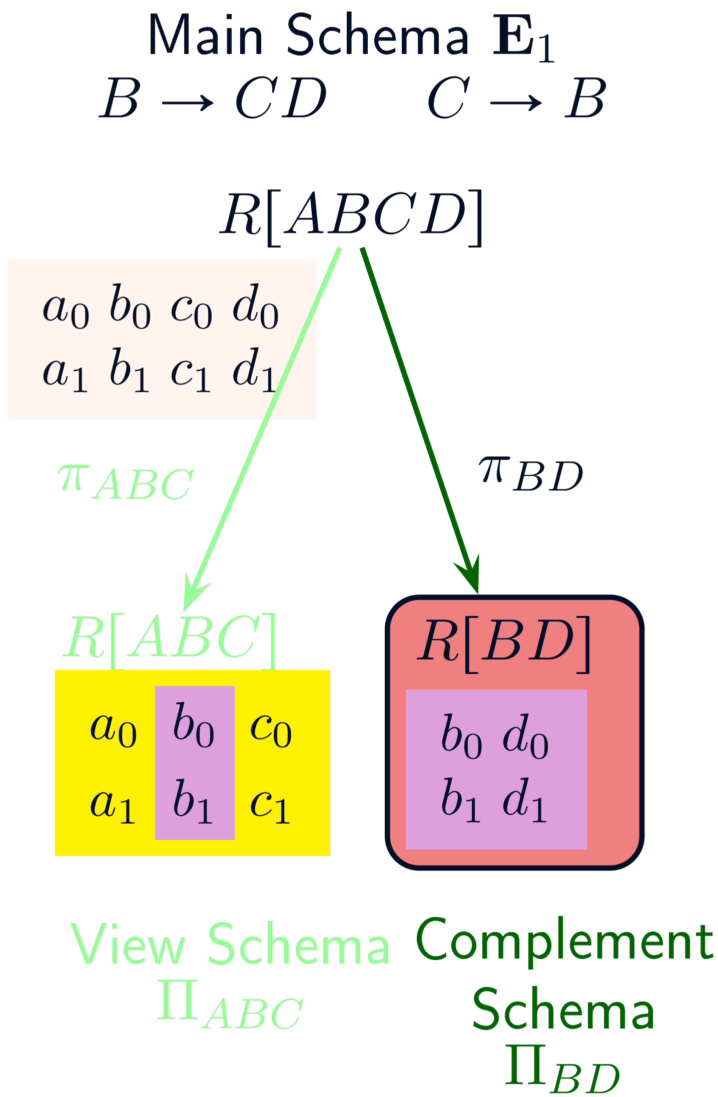
- It is useful to illustrate with a simple example.

- The view Π_{ABC} of the schema to the right has Π_{BD} as a natural monotonic meet complement.



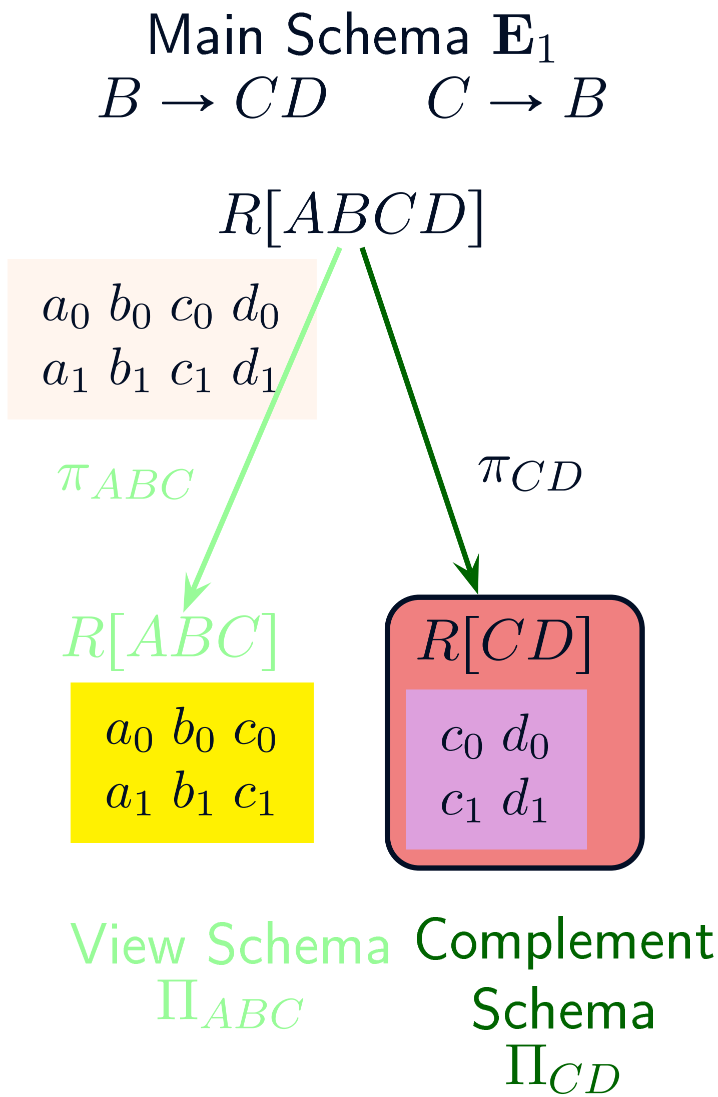
Incompatible View Updates

- The view Π_{ABC} of the schema to the right has Π_{BD} as a natural monotonic meet complement.
- With this complement, the allowable updates on Π_{ABC} are precisely those which keep Π_B constant.



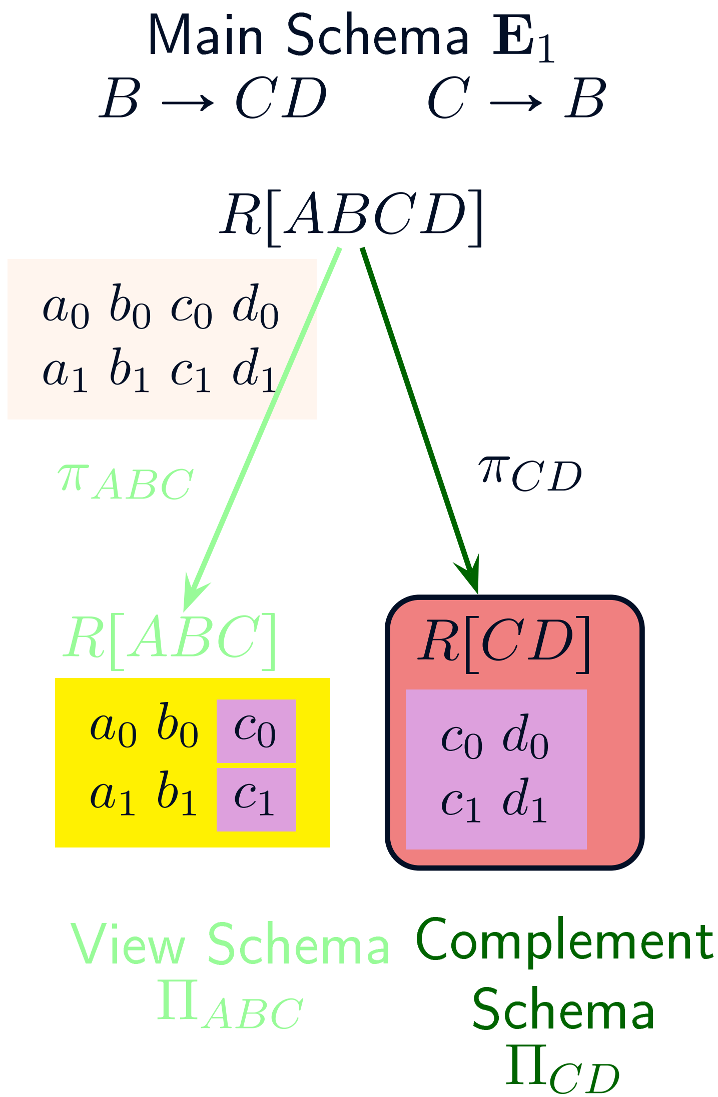
Incompatible View Updates

- The view Π_{ABC} of the schema to the right has Π_{BD} as a natural monotonic meet complement.
- With this complement, the allowable updates on Π_{ABC} are precisely those which keep Π_B constant.
- However, Π_{ABC} also has Π_{CD} as a natural meet complement.

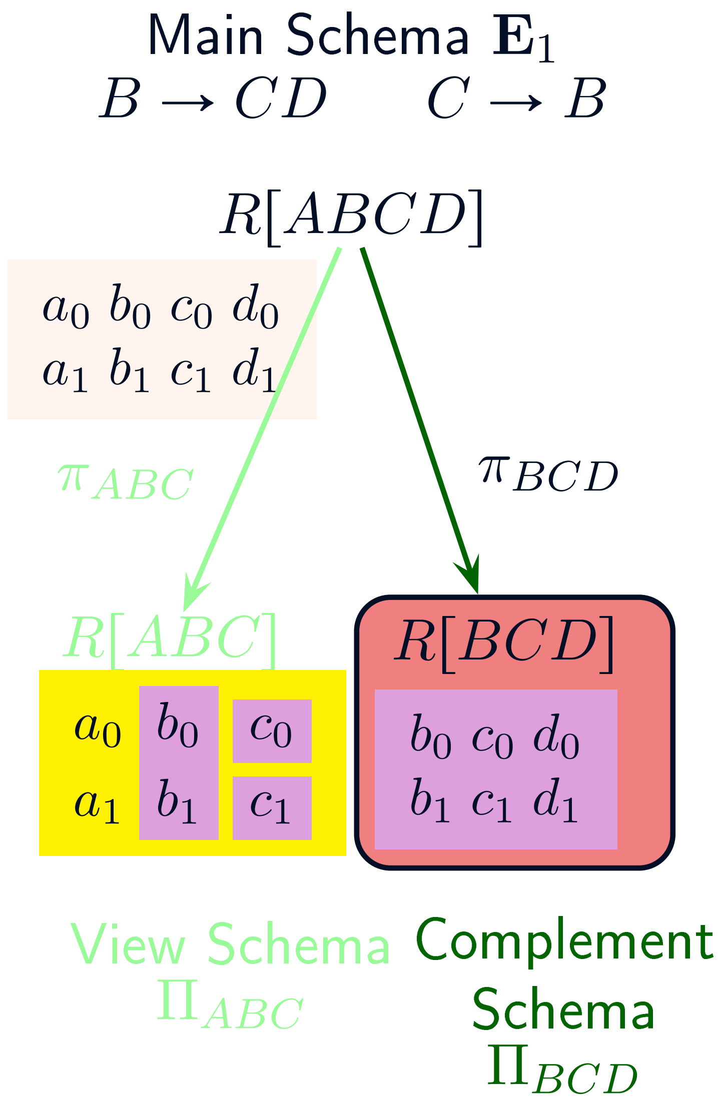


Incompatible View Updates

- The view Π_{ABC} of the schema to the right has Π_{BD} as a natural monotonic meet complement.
- With this complement, the allowable updates on Π_{ABC} are precisely those which keep Π_B constant.
- However, Π_{ABC} also has Π_{CD} as a natural meet complement.
- With this complement, the allowable updates on Π_{ABC} are precisely those which keep Π_C constant.

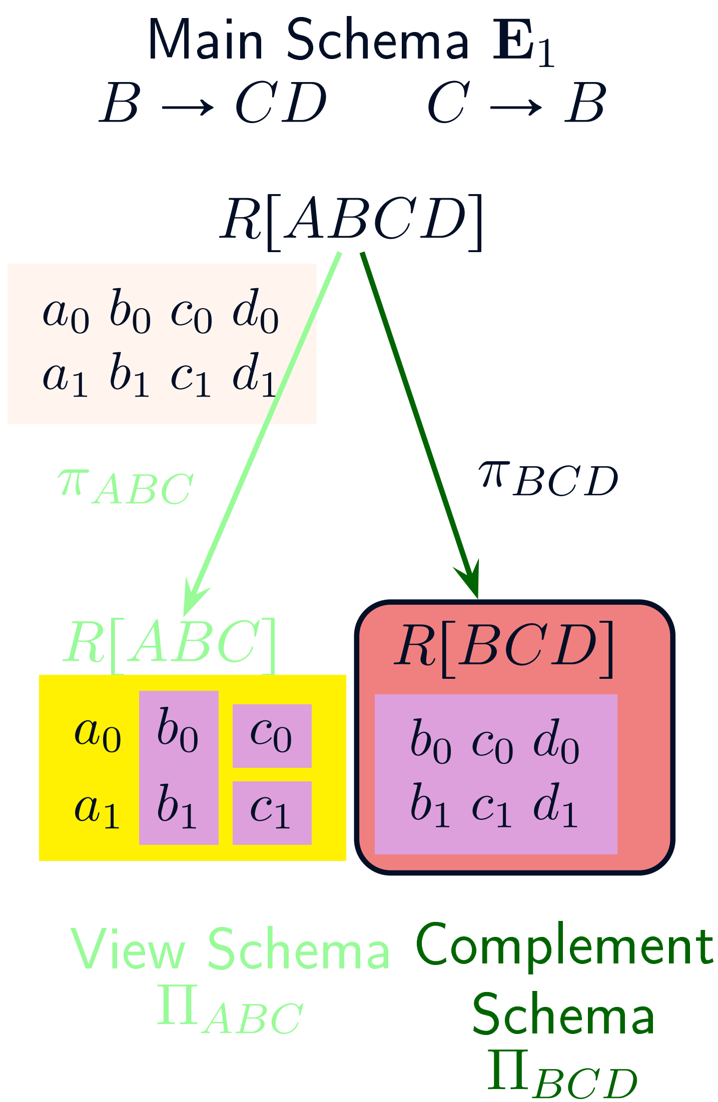


- The view Π_{ABC} of the schema to the right has Π_{BD} as a natural monotonic meet complement.
- With this complement, the allowable updates on Π_{ABC} are precisely those which keep Π_B constant.
- However, Π_{ABC} also has Π_{CD} as a natural meet complement.
- With this complement, the allowable updates on Π_{ABC} are precisely those which keep Π_C constant.
- The only updates allowable with both complements are those which hold Π_{BC} constant.

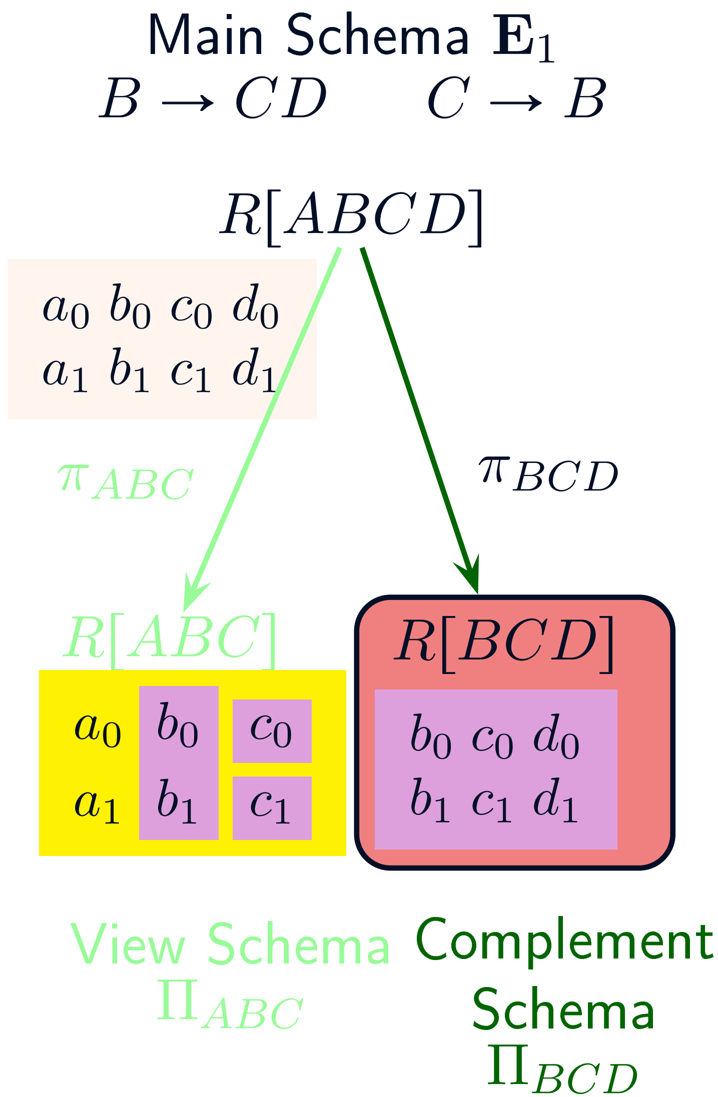


Incompatible View Updates

- The view Π_{ABC} of the schema to the right has Π_{BD} as a natural monotonic meet complement.
- With this complement, the allowable updates on Π_{ABC} are precisely those which keep Π_B constant.
- However, Π_{ABC} also has Π_{CD} as a natural meet complement.
- With this complement, the allowable updates on Π_{ABC} are precisely those which keep Π_C constant.
- The only updates allowable with both complements are those which hold Π_{BC} constant.
- The combined complement is effectively Π_{BCD}.



- The view Π_{ABC} of the schema to the right has Π_{BD} as a natural monotonic meet complement.
- With this complement, the allowable updates on Π_{ABC} are precisely those which keep Π_{B} constant.
- However, Π_{ABC} also has Π_{CD} as a natural meet complement.
- With this complement, the allowable updates on Π_{ABC} are precisely those which keep Π_{C} constant.
- The only updates allowable with both complements are those which hold Π_{BC} constant.
- The combined complement is effectively Π_{BCD}.
- There is no Π-complement which is more general than Π_{BD} or Π_{CD}.



Comparison of Projections and Optimal Complements

Context: A universal relational schema $R[\mathbf{U}]$ constrained by some dependencies $\mathcal{F}$.

Comparison of Projections and Optimal Complements

Context: A universal relational schema $R[\mathbf{U}]$ constrained by some dependencies $\mathcal{F}$.

- A Π-*view* is defined by a single projection.

Comparison of Projections and Optimal Complements

Context: A universal relational schema $R[\mathbf{U}]$ constrained by some dependencies $\mathcal{F}$.

- A Π-*view* is defined by a single projection.

Notation: Π_W is the projection onto attribute set W.

Comparison of Projections and Optimal Complements

Context: A universal relational schema $R[\mathbf{U}]$ constrained by some dependencies $\mathcal{F}$.

- A Π-*view* is defined by a single projection.

Notation: Π_W is the projection onto attribute set W.

- Projective views may be compared via their attributes.

$$\Pi_{\mathbf{W}_1} \preceq \Pi_{\mathbf{W}_2} \text{ iff } \mathbf{W}_1 \subseteq \mathbf{W}_2 \qquad (\mathbf{W}_1, \mathbf{W}_2 \subseteq \mathbf{U})$$

Comparison of Projections and Optimal Complements

Context: A universal relational schema $R[\mathbf{U}]$ constrained by some dependencies $\mathcal{F}$.

- A Π-*view* is defined by a single projection.

Notation: Π_W is the projection onto attribute set W.

- Projective views may be compared via their attributes.

$$\Pi_{\mathbf{W}_1} \preceq \Pi_{\mathbf{W}_2} \text{ iff } \mathbf{W}_1 \subseteq \mathbf{W}_2 \qquad (\mathbf{W}_1, \mathbf{W}_2 \subseteq \mathbf{U})$$

- Given a projective view $\Pi_{\mathbf{W}}$, a complement $\Pi_{\mathbf{W}'}$ is

Comparison of Projections and Optimal Complements

Context: A universal relational schema $R[\mathbf{U}]$ constrained by some dependencies $\mathcal{F}$.

- A Π-*view* is defined by a single projection.

Notation: Π_W is the projection onto attribute set W.

- Projective views may be compared via their attributes.

$$\Pi_{\mathbf{W}_1} \preceq \Pi_{\mathbf{W}_2} \text{ iff } \mathbf{W}_1 \subseteq \mathbf{W}_2 \qquad (\mathbf{W}_1, \mathbf{W}_2 \subseteq \mathbf{U})$$

- Given a projective view $\Pi_{\mathbf{W}}$, a complement $\Pi_{\mathbf{W}'}$ is
  - *minimal* if for no other complement $\Pi_{\mathbf{W}''}$ is it the case that $\mathbf{W}'' \subseteq \mathbf{W}'$;

Comparison of Projections and Optimal Complements

Context: A universal relational schema $R[\mathbf{U}]$ constrained by some dependencies $\mathcal{F}$.

- A Π-*view* is defined by a single projection.

Notation: Π_W is the projection onto attribute set W.

- Projective views may be compared via their attributes.

$$\Pi_{\mathbf{W}_1} \preceq \Pi_{\mathbf{W}_2} \text{ iff } \mathbf{W}_1 \subseteq \mathbf{W}_2 \qquad (\mathbf{W}_1, \mathbf{W}_2 \subseteq \mathbf{U})$$

- Given a projective view $\Pi_{\mathbf{W}}$, a complement $\Pi_{\mathbf{W}'}$ is
  - *minimal* if for no other complement $\Pi_{\mathbf{W}''}$ is it the case that $\mathbf{W}'' \subseteq \mathbf{W}'$;
  - *optimal* if for every other complement $\Pi_{\mathbf{W}''}$ it is the case that $\mathbf{W}' \subseteq \mathbf{W}''$.

A Context for Optimal Complements of Projections

- Let $R[\mathbf{U}]$ be universal relational schema constrained by some dependencies $\mathcal{F}$.

A Context for Optimal Complements of Projections

- Let $R[\mathbf{U}]$ be universal relational schema constrained by some dependencies $\mathcal{F}$.
- A *governing* JD is a representation of all JDs which hold on the schema.

A Context for Optimal Complements of Projections

- Let $R[\mathbf{U}]$ be universal relational schema constrained by some dependencies $\mathcal{F}$.
- A *governing* JD is a representation of all JDs which hold on the schema.
- More precisely, call a join dependency (JD) $\bowtie[\mathbf{U}_1, \ldots, \mathbf{U}_k]$ on $R[\mathbf{U}]$ *governing* (w.r.t. $\mathcal{F}$) if it defines a lossless decomposition of $R[\mathbf{U}]$ satisfying the following properties:

A Context for Optimal Complements of Projections

- Let $R[\mathbf{U}]$ be universal relational schema constrained by some dependencies $\mathcal{F}$.
- A *governing* JD is a representation of all JDs which hold on the schema.
- More precisely, call a join dependency (JD) $\bowtie[\mathbf{U}_1, \ldots, \mathbf{U}_k]$ on $R[\mathbf{U}]$ *governing* (w.r.t. $\mathcal{F}$) if it defines a lossless decomposition of $R[\mathbf{U}]$ satisfying the following properties:
    - *full*: $\mathbf{U}_1 \cup \ldots \cup \mathbf{U}_k = \mathbf{U}$;

- Let $R[\mathbf{U}]$ be universal relational schema constrained by some dependencies $\mathcal{F}$.
- A *governing* JD is a representation of all JDs which hold on the schema.
- More precisely, call a join dependency (JD) $\bowtie [\mathbf{U}_1, \ldots, \mathbf{U}_k]$ on $R[\mathbf{U}]$ *governing* (w.r.t. $\mathcal{F}$) if it defines a lossless decomposition of $R[\mathbf{U}]$ satisfying the following properties:
    - *full*: $\mathbf{U}_1 \cup \ldots \cup \mathbf{U}_k = \mathbf{U}$;
    - *entailed*: $\mathcal{F} \models \bowtie [\mathbf{U}_1, \ldots, \mathbf{U}_k]$;

A Context for Optimal Complements of Projections

- Let $R[\mathbf{U}]$ be universal relational schema constrained by some dependencies $\mathcal{F}$.
- A *governing* JD is a representation of all JDs which hold on the schema.
- More precisely, call a join dependency (JD) $\bowtie[\mathbf{U}_1, \ldots, \mathbf{U}_k]$ on $R[\mathbf{U}]$ *governing* (w.r.t. $\mathcal{F}$) if it defines a lossless decomposition of $R[\mathbf{U}]$ satisfying the following properties:
  - *full*: $\mathbf{U}_1 \cup \ldots \cup \mathbf{U}_k = \mathbf{U}$;
  - *entailed*: $\mathcal{F} \models \bowtie[\mathbf{U}_1, \ldots, \mathbf{U}_k]$;
  - *covering*: $\bowtie[\mathbf{U}_1, \ldots, \mathbf{U}_k] \models \varphi$ for every entailed JD (full or embedded) φ.

A Context for Optimal Complements of Projections

- Let $R[\mathbf{U}]$ be universal relational schema constrained by some dependencies $\mathcal{F}$.
- A *governing* JD is a representation of all JDs which hold on the schema.
- More precisely, call a join dependency (JD) $\bowtie [\mathbf{U}_1, \ldots, \mathbf{U}_k]$ on $R[\mathbf{U}]$ *governing* (w.r.t. $\mathcal{F}$) if it defines a lossless decomposition of $R[\mathbf{U}]$ satisfying the following properties:
  - *full*: $\mathbf{U}_1 \cup \ldots \cup \mathbf{U}_k = \mathbf{U}$;
  - *entailed*: $\mathcal{F} \models \ \bowtie [\mathbf{U}_1, \ldots, \mathbf{U}_k]$;
  - *covering*: $\bowtie [\mathbf{U}_1, \ldots, \mathbf{U}_k] \models \varphi$ for every entailed JD (full or embedded) φ.

Example: For $(R[ABCD]$ with $\mathcal{F} = \{B \to CD, C \to B\}$,
the JD $\bowtie [ABC, CD, BD]$ is governing.

A Context for Optimal Complements of Projections

- Let $R[\mathbf{U}]$ be universal relational schema constrained by some dependencies $\mathcal{F}$.
- A *governing* JD is a representation of all JDs which hold on the schema.
- More precisely, call a join dependency (JD) $\bowtie [\mathbf{U}_1, \ldots, \mathbf{U}_k]$ on $R[\mathbf{U}]$ *governing* (w.r.t. $\mathcal{F}$) if it defines a lossless decomposition of $R[\mathbf{U}]$ satisfying the following properties:
  - *full*: $\mathbf{U}_1 \cup \ldots \cup \mathbf{U}_k = \mathbf{U}$;
  - *entailed*: $\mathcal{F} \models \bowtie [\mathbf{U}_1, \ldots, \mathbf{U}_k]$;
  - *covering*: $\bowtie [\mathbf{U}_1, \ldots, \mathbf{U}_k] \models \varphi$ for every entailed JD (full or embedded) φ.

Example: For $(R[ABCD]$ with $\mathcal{F} = \{B \to CD, C \to B\}$,
the JD $\bowtie [ABC, CD, BD]$ is governing.

Example: For $(R[ABCD]$ with $\mathcal{F} = \{\bowtie [AB, BC]\}$, there is no (nontrivial) governing JD.

- To address the non-uniqueness of complements illustrated in examples, the following condition is introduced for the JD $\bowtie [\mathbf{U}_1, \ldots, \mathbf{U}_k]$:

- To address the non-uniqueness of complements illustrated in examples, the following condition is introduced for the JD $\bowtie [\mathbf{U}_1, \ldots, \mathbf{U}_k]$:
- *nonredundant*: for no proper $J \subsetneq \{\mathbf{U}_1, \ldots, \mathbf{U}_k\}$ is $\bowtie [J]$ both entailed and full.

Normalization and Nonredundancy

- To address the non-uniqueness of complements illustrated in examples, the following condition is introduced for the JD $\bowtie [\mathbf{U}_1, \ldots, \mathbf{U}_k]$:
- *nonredundant*: for no proper $J \subsetneq \{\mathbf{U}_1, \ldots, \mathbf{U}_k\}$ is $\bowtie [J]$ both entailed and full.
- There are two flavors of redundancy:

- To address the non-uniqueness of complements illustrated in examples, the following condition is introduced for the JD $\bowtie [\mathbf{U}_1, \ldots, \mathbf{U}_k]$:
- *nonredundant*: for no proper $J \subsetneq \{\mathbf{U}_1, \ldots, \mathbf{U}_k\}$ is $\bowtie [J]$ both entailed and full.
- There are two flavors of redundancy:
  - $\bowtie [\mathbf{U}_1, \ldots, \mathbf{U}_k]$ is *trivially redundant* (not *normalized*) if $\mathbf{U}_i \subsetneq \mathbf{U}_j$ for some distinct i, j.

Normalization and Nonredundancy

- To address the non-uniqueness of complements illustrated in examples, the following condition is introduced for the JD $\bowtie [\mathbf{U}_1, \ldots, \mathbf{U}_k]$:
- *nonredundant*: for no proper $J \subsetneq \{\mathbf{U}_1, \ldots, \mathbf{U}_k\}$ is $\bowtie [J]$ both entailed and full.
- There are two flavors of redundancy:
  - $\bowtie [\mathbf{U}_1, \ldots, \mathbf{U}_k]$ is *trivially redundant* (not *normalized*) if $\mathbf{U}_i \subsetneq \mathbf{U}_j$ for some distinct i, j.
  - This flavor of redundancy is "trivial" in the sense that it can be detected without any further knowledge of the underlying dependencies.

Normalization and Nonredundancy

- To address the non-uniqueness of complements illustrated in examples, the following condition is introduced for the JD $\bowtie[\mathbf{U}_1, \ldots, \mathbf{U}_k]$:
- *nonredundant*: for no proper $J \subsetneq \{\mathbf{U}_1, \ldots, \mathbf{U}_k\}$ is $\bowtie[J]$ both entailed and full.
- There are two flavors of redundancy:
  - $\bowtie[\mathbf{U}_1, \ldots, \mathbf{U}_k]$ is *trivially redundant* (not *normalized*) if $\mathbf{U}_i \subsetneq \mathbf{U}_j$ for some distinct i, j.
  - This flavor of redundancy is "trivial" in the sense that it can be detected without any further knowledge of the underlying dependencies.
  - It may always be removed without changing the semantics.

Normalization and Nonredundancy

- To address the non-uniqueness of complements illustrated in examples, the following condition is introduced for the JD $\bowtie [\mathbf{U}_1, \ldots, \mathbf{U}_k]$:
- *nonredundant*: for no proper $J \subsetneq \{\mathbf{U}_1, \ldots, \mathbf{U}_k\}$ is $\bowtie [J]$ both entailed and full.
- There are two flavors of redundancy:
  - $\bowtie [\mathbf{U}_1, \ldots, \mathbf{U}_k]$ is *trivially redundant* (not *normalized*) if $\mathbf{U}_i \subsetneq \mathbf{U}_j$ for some distinct i, j.
  - This flavor of redundancy is "trivial" in the sense that it can be detected without any further knowledge of the underlying dependencies.
  - It may always be removed without changing the semantics.

  Example: $\bowtie [AC, ABC, CD]$ is trivially redundant.

Normalization and Nonredundancy

- To address the non-uniqueness of complements illustrated in examples, the following condition is introduced for the JD $\bowtie [\mathbf{U}_1, \ldots, \mathbf{U}_k]$:
- *nonredundant*: for no proper $J \subsetneq \{\mathbf{U}_1, \ldots, \mathbf{U}_k\}$ is $\bowtie [J]$ both entailed and full.
- There are two flavors of redundancy:
  - $\bowtie [\mathbf{U}_1, \ldots, \mathbf{U}_k]$ is *trivially redundant* (not *normalized*) if $\mathbf{U}_i \subsetneq \mathbf{U}_j$ for some distinct i, j.
  - This flavor of redundancy is "trivial" in the sense that it can be detected without any further knowledge of the underlying dependencies.
  - It may always be removed without changing the semantics.

  Example: $\bowtie [AC, ABC, CD]$ is trivially redundant.
  - Otherwise, redundancy is *essential* and must be determined by examining the underlying dependencies.

Normalization and Nonredundancy

- To address the non-uniqueness of complements illustrated in examples, the following condition is introduced for the JD $\bowtie [\mathbf{U}_1, \ldots, \mathbf{U}_k]$:
- *nonredundant*: for no proper $J \subsetneq \{\mathbf{U}_1, \ldots, \mathbf{U}_k\}$ is $\bowtie [J]$ both entailed and full.
- There are two flavors of redundancy:
  - $\bowtie [\mathbf{U}_1, \ldots, \mathbf{U}_k]$ is *trivially redundant* (not *normalized*) if $\mathbf{U}_i \subsetneq \mathbf{U}_j$ for some distinct i, j.
  - This flavor of redundancy is "trivial" in the sense that it can be detected without any further knowledge of the underlying dependencies.
  - It may always be removed without changing the semantics.

  Example: $\bowtie [AC, ABC, CD]$ is trivially redundant.
  - Otherwise, redundancy is *essential* and must be determined by examining the underlying dependencies.

  Example: For $R[ABCD]$ with $\mathcal{F} = \{B \to CD, C \to B\}$,
  the JD $\bowtie [ABC, CD, BD]$ is governing but essentially redundant,
  since $\bowtie [ABC, CD]$ (as well as $\bowtie [ABC, BD]$) is both entailed and full.

Context: Universal relational schema $R[\mathbf{U}]$ constrained by some dependencies $\mathcal{F}$.
Nonredundant governing JD $\bowtie [\mathcal{A}]$ with $\mathcal{A} \stackrel{\text{def}}{=} \{\mathbf{U}_i \mid 1 \leqslant i \leqslant k\}$.

Characterization of Optimal Π-Complements

Context: Universal relational schema $R[\mathbf{U}]$ constrained by some dependencies $\mathcal{F}$.
Nonredundant governing JD $\bowtie[\mathcal{A}]$ with $\mathcal{A} \stackrel{\text{def}}{=} \{\mathbf{U}_i \mid 1 \leqslant i \leqslant k\}$.

Theorem: Let $\mathbf{W} \subseteq \mathbf{U}$, and define $\mathbf{W}' = \bigcup\{\mathbf{U}_i \in \mathcal{A} \mid \mathbf{U}_i \nsubseteq \mathbf{W}\}$.
Then $\Pi_{\mathbf{W}'}$ is an optimal Π-complement of $\Pi_{\mathbf{W}}$. If the JD is dependency preserving, then it is furthermore a meet complement. □

Characterization of Optimal Π-Complements

Context: Universal relational schema $R[\mathbf{U}]$ constrained by some dependencies $\mathcal{F}$.
Nonredundant governing JD $\bowtie[\mathcal{A}]$ with $\mathcal{A} \stackrel{\text{def}}{=} \{\mathbf{U}_i \mid 1 \leqslant i \leqslant k\}$.

Theorem: Let $\mathbf{W} \subseteq \mathbf{U}$, and define $\mathbf{W}' = \bigcup\{\mathbf{U}_i \in \mathcal{A} \mid \mathbf{U}_i \nsubseteq \mathbf{W}\}$.
Then $\Pi_{\mathbf{W}'}$ is an optimal Π-complement of $\Pi_{\mathbf{W}}$. If the JD is dependency preserving, then it is furthermore a meet complement. $\square$

Example context: $R[ABCDE]$ $C \rightarrow D$ $D \rightarrow E$ $\models$
$\bowtie[ABC, CD, DE]$ governing and nonredundant.

Context: Universal relational schema $R[\mathbf{U}]$ constrained by some dependencies $\mathcal{F}$.
Nonredundant governing JD $\bowtie[\mathcal{A}]$ with $\mathcal{A} \stackrel{\text{def}}{=} \{\mathbf{U}_i \mid 1 \leqslant i \leqslant k\}$.

Theorem: Let $\mathbf{W} \subseteq \mathbf{U}$, and define $\mathbf{W}' = \bigcup\{\mathbf{U}_i \in \mathcal{A} \mid \mathbf{U}_i \nsubseteq \mathbf{W}\}$.
Then $\Pi_{\mathbf{W}'}$ is an optimal Π-complement of $\Pi_{\mathbf{W}}$. If the JD is dependency preserving, then it is furthermore a meet complement. $\square$

Example context: $R[ABCDE]$ $C \to D$ $D \to E$ $\models$
$\bowtie[ABC, CD, DE]$ governing and nonredundant.

- The optimal Π-complement of Π_{ABC} is $\Pi_{CD \cup DE} = \Pi_{CDE}$.

Characterization of Optimal Π-Complements

Context: Universal relational schema $R[\mathbf{U}]$ constrained by some dependencies $\mathcal{F}$.
Nonredundant governing JD $\bowtie[\mathcal{A}]$ with $\mathcal{A} \stackrel{\text{def}}{=} \{\mathbf{U}_i \mid 1 \leqslant i \leqslant k\}$.

Theorem: Let $\mathbf{W} \subseteq \mathbf{U}$, and define $\mathbf{W}' = \bigcup\{\mathbf{U}_i \in \mathcal{A} \mid \mathbf{U}_i \nsubseteq \mathbf{W}\}$.
Then $\Pi_{\mathbf{W}'}$ is an optimal Π-complement of $\Pi_{\mathbf{W}}$. If the JD is dependency preserving, then it is furthermore a meet complement. $\square$

Example context: $R[ABCDE]$ $C \rightarrow D$ $D \rightarrow E$ $\models$
$\bowtie[ABC, CD, DE]$ governing and nonredundant.

- The optimal Π-complement of Π_{ABC} is $\Pi_{CD \cup DE} = \Pi_{CDE}$.
- The optimal Π-complement of Π_{ABCE} is $\Pi_{CD \cup DE} = \Pi_{CDE}$ also.

Context: Universal relational schema $R[\mathbf{U}]$ constrained by some dependencies $\mathcal{F}$.
Nonredundant governing JD $\bowtie [\mathcal{A}]$ with $\mathcal{A} \stackrel{\text{def}}{=} \{\mathbf{U}_i \mid 1 \leqslant i \leqslant k\}$.

Theorem: Let $\mathbf{W} \subseteq \mathbf{U}$, and define $\mathbf{W}' = \bigcup\{\mathbf{U}_i \in \mathcal{A} \mid \mathbf{U}_i \nsubseteq \mathbf{W}\}$.
Then $\Pi_{\mathbf{W}'}$ is an optimal Π-complement of $\Pi_{\mathbf{W}}$. If the JD is dependency preserving, then it is furthermore a meet complement. $\square$

Example context: $R[ABCDE]$ $C \rightarrow D$ $D \rightarrow E$ $\models$
$\bowtie [ABC, CD, DE]$ governing and nonredundant.

- The optimal Π-complement of Π_{ABC} is $\Pi_{CD \cup DE} = \Pi_{CDE}$.
- The optimal Π-complement of Π_{ABCE} is $\Pi_{CD \cup DE} = \Pi_{CDE}$ also.
- The optimal Π-complement of Π_{ABCD} is Π_{DE}.

Characterization of Optimal Π-Complements

Context: Universal relational schema $R[\mathbf{U}]$ constrained by some dependencies $\mathcal{F}$.
Nonredundant governing JD $\bowtie[\mathcal{A}]$ with $\mathcal{A} \stackrel{\text{def}}{=} \{\mathbf{U}_i \mid 1 \leqslant i \leqslant k\}$.

Theorem: Let $\mathbf{W} \subseteq \mathbf{U}$, and define $\mathbf{W}' = \bigcup\{\mathbf{U}_i \in \mathcal{A} \mid \mathbf{U}_i \nsubseteq \mathbf{W}\}$.
Then $\Pi_{\mathbf{W}'}$ is an optimal Π-complement of $\Pi_{\mathbf{W}}$. If the JD is dependency preserving, then it is furthermore a meet complement. $\square$

Example context: $R[ABCDE]$ $C \to D$ $D \to E$ $\models$
$\bowtie[ABC, CD, DE]$ governing and nonredundant.

- The optimal Π-complement of Π_{ABC} is $\Pi_{CD \cup DE} = \Pi_{CDE}$.
- The optimal Π-complement of Π_{ABCE} is $\Pi_{CD \cup DE} = \Pi_{CDE}$ also.
- The optimal Π-complement of Π_{ABCD} is Π_{DE}.
- The optimal Π-complement of Π_{AB} is $\Pi_{ABC \cup CD \cup DE} = \Pi_{ABCDE}$.

Characterization of Optimal Π-Complements

Context: Universal relational schema $R[\mathbf{U}]$ constrained by some dependencies $\mathcal{F}$.
Nonredundant governing JD $\bowtie [\mathcal{A}]$ with $\mathcal{A} \stackrel{\text{def}}{=} \{\mathbf{U}_i \mid 1 \leqslant i \leqslant k\}$.

Theorem: Let $\mathbf{W} \subseteq \mathbf{U}$, and define $\mathbf{W}' = \bigcup\{\mathbf{U}_i \in \mathcal{A} \mid \mathbf{U}_i \nsubseteq \mathbf{W}\}$.
Then $\Pi_{\mathbf{W}'}$ is an optimal Π-complement of $\Pi_{\mathbf{W}}$. If the JD is dependency preserving, then it is furthermore a meet complement. □

Example context: $R[ABCDE]$ $C \to D$ $D \to E$ $\models$
$\bowtie [ABC, CD, DE]$ governing and nonredundant.

- The optimal Π-complement of Π_{ABC} is $\Pi_{CD \cup DE} = \Pi_{CDE}$.
- The optimal Π-complement of Π_{ABCE} is $\Pi_{CD \cup DE} = \Pi_{CDE}$ also.
- The optimal Π-complement of Π_{ABCD} is Π_{DE}.
- The optimal Π-complement of Π_{AB} is $\Pi_{ABC \cup CD \cup DE} = \Pi_{ABCDE}$.
- The optimal Π-complement of Π_{CD} is $\Pi_{ABC \cup DE} = \Pi_{ABCDE}$ also.

Example context continued: $R[ABCDE]$ $C \rightarrow D$ $D \rightarrow E$ $\models$
$\bowtie [ABC, CD, DE]$ governing and nonredundant.

Example context continued: $R[ABCDE]$ $C \to D$ $D \to E$ $\models$
$\bowtie [ABC, CD, DE]$ governing and nonredundant.

- For Π_{CD}, the optimal Π-complement Π_{ABCDE} allows no updates at all under the constant-complement strategy.

Example context continued: $R[ABCDE]\ C \rightarrow D\ \ D \rightarrow E\ \models$
$\bowtie [ABC, CD, DE]$ governing and nonredundant.

- For Π_{CD}, the optimal Π-complement Π_{ABCDE} allows no updates at all under the constant-complement strategy.
- However, some updates are clearly possible.

An Issue of Suboptimality within a Wider Context

Example context continued: $R[ABCDE]\ C \to D\ \ D \to E\ \models$
$\bowtie [ABC, CD, DE]$ governing and nonredundant.

- For Π_{CD}, the optimal Π-complement Π_{ABCDE} allows no updates at all under the constant-complement strategy.
- However, some updates are clearly possible.
- Let $M = \{R(\mathrm{a}_1, \mathrm{b}_1, \mathrm{c}_1, \mathrm{d}_1, \mathrm{e}_1), R(\mathrm{a}_2, \mathrm{b}_2, \mathrm{c}_2, \mathrm{d}_2, \mathrm{e}_2)\}$ be the current state of the main schema.

An Issue of Suboptimality within a Wider Context

Example context continued: $R[ABCDE]$ $C \to D$ $D \to E$ $\models$ $\bowtie [ABC, CD, DE]$ governing and nonredundant.

- For Π_{CD}, the optimal Π-complement Π_{ABCDE} allows no updates at all under the constant-complement strategy.
- However, some updates are clearly possible.
- Let $M = \{R(\mathrm{a}_1, \mathrm{b}_1, \mathrm{c}_1, \mathrm{d}_1, \mathrm{e}_1), R(\mathrm{a}_2, \mathrm{b}_2, \mathrm{c}_2, \mathrm{d}_2, \mathrm{e}_2)\}$ be the current state of the main schema.
- Consider the update (N, N') to Π_{CD} with $N = \{R(\mathrm{c}_1, \mathrm{d}_1), R(\mathrm{c}_2, \mathrm{d}_2)\}$. and $N' = \{R(\mathrm{c}_1, \mathrm{d}_2), R(\mathrm{c}_2, \mathrm{d}_1)\}$.

An Issue of Suboptimality within a Wider Context

Example context continued: $R[ABCDE]\ C \to D\ \ D \to E\ \models$
$\bowtie [ABC, CD, DE]$ governing and nonredundant.

- For Π_{CD}, the optimal Π-complement Π_{ABCDE} allows no updates at all under the constant-complement strategy.
- However, some updates are clearly possible.
- Let $M = \{R(\mathrm{a}_1, \mathrm{b}_1, \mathrm{c}_1, \mathrm{d}_1, \mathrm{e}_1), R(\mathrm{a}_2, \mathrm{b}_2, \mathrm{c}_2, \mathrm{d}_2, \mathrm{e}_2)\}$ be the current state of the main schema.
- Consider the update (N, N') to Π_{CD} with $N = \{R(\mathrm{c}_1, \mathrm{d}_1), R(\mathrm{c}_2, \mathrm{d}_2)\}$. and $N' = \{R(\mathrm{c}_1, \mathrm{d}_2), R(\mathrm{c}_2, \mathrm{d}_1)\}$.
- The reflection $M' = \{R(\mathrm{a}_1, \mathrm{b}_1, \mathrm{c}_1, \mathrm{d}_2, \mathrm{e}_2) R(\mathrm{a}_2, \mathrm{b}_2, \mathrm{c}_2, \mathrm{d}_1, \mathrm{e}_1)\}$ keeps both Π_{ABC} and Π_{CD} constant.

An Issue of Suboptimality within a Wider Context

Example context continued: $R[ABCDE]\ C \to D\ \ D \to E\ \models$
$\bowtie [ABC, CD, DE]$ governing and nonredundant.

- For Π_{CD}, the optimal Π-complement Π_{ABCDE} allows no updates at all under the constant-complement strategy.
- However, some updates are clearly possible.
- Let $M = \{R(\mathrm{a}_1, \mathrm{b}_1, \mathrm{c}_1, \mathrm{d}_1, \mathrm{e}_1), R(\mathrm{a}_2, \mathrm{b}_2, \mathrm{c}_2, \mathrm{d}_2, \mathrm{e}_2)\}$ be the current state of the main schema.
- Consider the update (N, N') to Π_{CD} with $N = \{R(\mathrm{c}_1, \mathrm{d}_1), R(\mathrm{c}_2, \mathrm{d}_2)\}$. and $N' = \{R(\mathrm{c}_1, \mathrm{d}_2), R(\mathrm{c}_2, \mathrm{d}_1)\}$.
- The reflection $M' = \{R(\mathrm{a}_1, \mathrm{b}_1, \mathrm{c}_1, \mathrm{d}_2, \mathrm{e}_2) R(\mathrm{a}_2, \mathrm{b}_2, \mathrm{c}_2, \mathrm{d}_1, \mathrm{e}_1)\}$ keeps both Π_{ABC} and Π_{CD} constant.
- The view $\Pi_{ABC} \vee \Pi_{DE}$ which contains two projections, $R[ABC]$ and $R[DE]$, is a complement of Π_{CD}.

An Issue of Suboptimality within a Wider Context

Example context continued: $R[ABCDE]\ C \to D\ \ D \to E\ \models$
$\bowtie [ABC, CD, DE]$ governing and nonredundant.

- For Π_{CD}, the optimal Π-complement Π_{ABCDE} allows no updates at all under the constant-complement strategy.
- However, some updates are clearly possible.
- Let $M = \{R(\mathrm{a}_1, \mathrm{b}_1, \mathrm{c}_1, \mathrm{d}_1, \mathrm{e}_1), R(\mathrm{a}_2, \mathrm{b}_2, \mathrm{c}_2, \mathrm{d}_2, \mathrm{e}_2)\}$ be the current state of the main schema.
- Consider the update (N, N') to Π_{CD} with $N = \{R(\mathrm{c}_1, \mathrm{d}_1), R(\mathrm{c}_2, \mathrm{d}_2)\}$. and $N' = \{R(\mathrm{c}_1, \mathrm{d}_2), R(\mathrm{c}_2, \mathrm{d}_1)\}$.
- The reflection $M' = \{R(\mathrm{a}_1, \mathrm{b}_1, \mathrm{c}_1, \mathrm{d}_2, \mathrm{e}_2)R(\mathrm{a}_2, \mathrm{b}_2, \mathrm{c}_2, \mathrm{d}_1, \mathrm{e}_1)\}$ keeps both Π_{ABC} and Π_{CD} constant.
- The view $\Pi_{ABC} \vee \Pi_{DE}$ which contains two projections, $R[ABC]$ and $R[DE]$, is a complement of Π_{CD}.
- Thus, (M, M') is a constant-complement reflection of (N, N') with complement $\Pi_{ABC} \vee \Pi_{DE}$.

Complements of $\bigvee\Pi$-views

Context: Universal relational schema $R[\mathbf{U}]$ constrained by some dependencies $\mathcal{F}$; Nonredundant governing JD $\bowtie[\mathcal{A}]$ with $\mathcal{A} \overset{\text{def}}{=} \{\mathbf{U}_i \mid 1 \leqslant i \leqslant k\}$.

Complements of $\bigvee\Pi$-views

Context: Universal relational schema $R[\mathbf{U}]$ constrained by some dependencies $\mathcal{F}$;
Nonredundant governing JD $\bowtie[\mathcal{A}]$ with $\mathcal{A} \stackrel{\text{def}}{=} \{\mathbf{U}_i \mid 1 \leqslant i \leqslant k\}$.

- A $\bigvee\Pi$-view is defined by a set of projections on a (universal) relational schema.

Context: Universal relational schema $R[\mathbf{U}]$ constrained by some dependencies $\mathcal{F}$; Nonredundant governing JD $\bowtie[\mathcal{A}]$ with $\mathcal{A} \stackrel{\text{def}}{=} \{\mathbf{U}_i \mid 1 \leqslant i \leqslant k\}$.

- A $\bigvee\Pi$-view is defined by a set of projections on a (universal) relational schema.

Example and notation: $\Pi_{ABC} \vee \Pi_{DE} = \bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \{ABC, DE\}\}$.

Context: Universal relational schema $R[\mathbf{U}]$ constrained by some dependencies $\mathcal{F}$; Nonredundant governing JD $\bowtie [\mathcal{A}]$ with $\mathcal{A} \stackrel{\text{def}}{=} \{\mathbf{U}_i \mid 1 \leqslant i \leqslant k\}$.

- A $\bigvee\Pi$-view is defined by a set of projections on a (universal) relational schema.

Example and notation: $\Pi_{ABC} \vee \Pi_{DE} = \bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \{ABC, DE\}\}$.

Theorem: For any partition $\{\mathcal{A}_1, \mathcal{A}_2\}$ of $\mathcal{A}$, $\bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{A}_1\}$ and $\bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{A}_2\}$ are complements. They are furthermore meet complements if the JD is dependency preserving. □

Complements of $\bigvee\Pi$-views

Context: Universal relational schema $R[\mathbf{U}]$ constrained by some dependencies $\mathcal{F}$;
Nonredundant governing JD $\bowtie[\mathcal{A}]$ with $\mathcal{A} \stackrel{\text{def}}{=} \{\mathbf{U}_i \mid 1 \leqslant i \leqslant k\}$.

- A $\bigvee\Pi$-view is defined by a set of projections on a (universal) relational schema.

Example and notation: $\Pi_{ABC} \vee \Pi_{DE} = \bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \{ABC, DE\}\}$.

Theorem: For any partition $\{\mathcal{A}_1, \mathcal{A}_2\}$ of $\mathcal{A}$, $\bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{A}_1\}$ and $\bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{A}_2\}$ are complements. They are furthermore meet complements if the JD is dependency preserving. $\square$

Example context: $R[ABCDE]$ $C \rightarrow D$ $D \rightarrow E$ $E \rightarrow F$ $\models$
$\bowtie[ABC, CD, DE, EF]$ is governing and nonredundant.

Complements of $\vee\Pi$-views

Context: Universal relational schema $R[\mathbf{U}]$ constrained by some dependencies $\mathcal{F}$;
Nonredundant governing JD $\bowtie [\mathcal{A}]$ with $\mathcal{A} \stackrel{\text{def}}{=} \{\mathbf{U}_i \mid 1 \leqslant i \leqslant k\}$.

- A $\vee\Pi$-view is defined by a set of projections on a (universal) relational schema.

Example and notation: $\Pi_{ABC} \vee \Pi_{DE} = \bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \{ABC, DE\}\}$.

Theorem: For any partition $\{\mathcal{A}_1, \mathcal{A}_2\}$ of $\mathcal{A}$, $\bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{A}_1\}$ and $\bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{A}_2\}$ are complements. They are furthermore meet complements if the JD is dependency preserving. $\square$

Example context: $R[ABCDE]\ C \to D\ \ D \to E\ \ E \to F\ \models$
$\bowtie [ABC, CD, DE, EF]$ is governing and nonredundant.

- $\Pi_{ABC} \vee \Pi_{DE}$ and $\Pi_{CD} \vee \Pi_{EF}$ are meet complements.

Complements of $\bigvee\Pi$-views

Context: Universal relational schema $R[\mathbf{U}]$ constrained by some dependencies $\mathcal{F}$;
Nonredundant governing JD $\bowtie [\mathcal{A}]$ with $\mathcal{A} \stackrel{\text{def}}{=} \{\mathbf{U}_i \mid 1 \leqslant i \leqslant k\}$.

- A $\bigvee\Pi$-view is defined by a set of projections on a (universal) relational schema.

Example and notation: $\Pi_{ABC} \vee \Pi_{DE} = \bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \{ABC, DE\}\}$.

Theorem: For any partition $\{\mathcal{A}_1, \mathcal{A}_2\}$ of $\mathcal{A}$, $\bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{A}_1\}$ and $\bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{A}_2\}$ are complements. They are furthermore meet complements if the JD is dependency preserving. $\square$

Example context: $R[ABCDE]$ $C \to D$ $D \to E$ $E \to F$ $\models$
$\bowtie [ABC, CD, DE, EF]$ is governing and nonredundant.

- $\Pi_{ABC} \vee \Pi_{DE}$ and $\Pi_{CD} \vee \Pi_{EF}$ are meet complements.
- Note that $\Pi_{ABC} \vee \Pi_{DE} \neq \Pi_{ABCDE}$, $\Pi_{CD} \vee \Pi_{EF} \neq \Pi_{CDEF}$; they are not even isomorphic.

Comparison of $\bigvee\Pi$-complements

Context: Universal relational schema $R[\mathbf{U}]$ constrained by some dependencies $\mathcal{F}$;
Nonredundant governing JD $\bowtie [\mathcal{A}]$ with $\mathcal{A} \stackrel{\text{def}}{=} \{\mathbf{U}_i \mid 1 \leqslant i \leqslant k\}$.

Comparison of $\bigvee\Pi$-complements

Context: Universal relational schema $R[\mathbf{U}]$ constrained by some dependencies $\mathcal{F}$; Nonredundant governing JD $\bowtie[\mathcal{A}]$ with $\mathcal{A} \stackrel{\text{def}}{=} \{\mathbf{U}_i \mid 1 \leqslant i \leqslant k\}$.

- A first attempt at a definition of comparison:

$$\bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{B}_1\} \preceq \bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{B}_2\} \quad \text{iff} \quad (\forall \mathbf{Y} \in \mathcal{B}_1)(\exists \mathbf{Y}' \in \mathcal{B}_2)(\mathbf{Y} \subseteq \mathbf{Y}').$$

Comparison of $\vee\Pi$-complements

Context: Universal relational schema $R[\mathbf{U}]$ constrained by some dependencies $\mathcal{F}$; Nonredundant governing JD $\bowtie[\mathcal{A}]$ with $\mathcal{A} \stackrel{\text{def}}{=} \{\mathbf{U}_i \mid 1 \leqslant i \leqslant k\}$.

- A first attempt at a definition of comparison:

$$\bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{B}_1\} \preceq \bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{B}_2\} \quad \text{iff} \quad (\forall \mathbf{Y} \in \mathcal{B}_1)(\exists \mathbf{Y}' \in \mathcal{B}_2)(\mathbf{Y} \subseteq \mathbf{Y}').$$

Counterexample: $R[ABCDE]\ C \to D\ \ D \to E\ \models\ \bowtie[ABC, CD, DE]$.
Since the embedded JD $\bowtie[ABC, CD]$ is implied, Π_{ABCD} is effectively the same as $\Pi_{ABC} \vee \Pi_{CD}$.

Comparison of $\vee\Pi$-complements

Context: Universal relational schema $R[\mathbf{U}]$ constrained by some dependencies $\mathcal{F}$; Nonredundant governing JD $\bowtie[\mathcal{A}]$ with $\mathcal{A} \stackrel{\text{def}}{=} \{\mathbf{U}_i \mid 1 \leqslant i \leqslant k\}$.

- A first attempt at a definition of comparison:

$$\bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{B}_1\} \preceq \bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{B}_2\} \quad \text{iff} \quad (\forall \mathbf{Y} \in \mathcal{B}_1)(\exists \mathbf{Y}' \in \mathcal{B}_2)(\mathbf{Y} \subseteq \mathbf{Y}').$$

Counterexample: $R[ABCDE]\ C \to D\ \ D \to E\ \models\ \bowtie[ABC, CD, DE]$.
Since the embedded JD $\bowtie[ABC, CD]$ is implied, Π_{ABCD} is effectively the same as $\Pi_{ABC} \vee \Pi_{CD}$.

- A better definition of comparison: every LHS attribute set is a subset of a valid join of a RHS set.

$$\bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{B}_1\} \preceq \bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{B}_2\} \quad \text{iff}$$

$$(\forall \mathbf{Y} \in \mathcal{B}_1)(\exists \mathcal{B}_3 \subseteq \mathcal{B}_2)((\bowtie[\mathcal{B}_3]\ \text{valid}) \wedge (\mathbf{Y} \subseteq \bigcup \mathcal{B}_3)).$$

Context: Universal relational schema $R[\mathbf{U}]$ constrained by some dependencies $\mathcal{F}$;
Nonredundant governing JD $\bowtie[\mathcal{A}]$ with $\mathcal{A} \stackrel{\text{def}}{=} \{\mathbf{U}_i \mid 1 \leqslant i \leqslant k\}$.

$$\bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{B}_1\} \preceq \bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{B}_2\} \quad \text{iff}$$

$$(\forall \mathbf{Y} \in \mathcal{B}_1)(\exists \mathcal{B}_3 \subseteq \mathcal{B}_2)((\bowtie[\mathcal{B}_3] \text{ valid}) \wedge (\mathbf{Y} \subseteq \bigcup \mathcal{B}_3)).$$

Context: Universal relational schema $R[\mathbf{U}]$ constrained by some dependencies $\mathcal{F}$;
Nonredundant governing JD $\bowtie[\mathcal{A}]$ with $\mathcal{A} \stackrel{\text{def}}{=} \{\mathbf{U}_i \mid 1 \leqslant i \leqslant k\}$.

$$\bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{B}_1\} \preceq \bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{B}_2\} \quad \text{iff}$$

$$(\forall \mathbf{Y} \in \mathcal{B}_1)(\exists \mathcal{B}_3 \subseteq \mathcal{B}_2)((\bowtie[\mathcal{B}_3] \text{ valid}) \wedge (\mathbf{Y} \subseteq \bigcup \mathcal{B}_3)).$$

- $\bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{B}_2\}$, is an *optimal* complement of $\bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{B}_1\}$ if
$$\bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{B}_2\} \preceq \bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{B}_3\}$$
for every other $\bigvee\Pi$-complement $\bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{B}_3\}$,

Optimal $\vee\Pi$-complements

Context: Universal relational schema $R[\mathbf{U}]$ constrained by some dependencies $\mathcal{F}$;
Nonredundant governing JD $\bowtie[\mathcal{A}]$ with $\mathcal{A} \stackrel{\text{def}}{=} \{\mathbf{U}_i \mid 1 \leqslant i \leqslant k\}$.

$$\bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{B}_1\} \preceq \bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{B}_2\} \quad \text{iff}$$
$$(\forall \mathbf{Y} \in \mathcal{B}_1)(\exists \mathcal{B}_3 \subseteq \mathcal{B}_2)((\bowtie[\mathcal{B}_3] \text{ valid}) \wedge (\mathbf{Y} \subseteq \bigcup \mathcal{B}_3)).$$

- $\bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{B}_2\}$, is an *optimal* complement of $\bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{B}_1\}$ if
$$\bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{B}_2\} \preceq \bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{B}_3\}$$
for every other $\vee\Pi$-complement $\bigvee\{\Pi_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{B}_3\}$,
- For $\mathbf{W} \subseteq \mathbf{U}$, define $\text{JCompl}\langle \mathbf{W}, \mathcal{A}\rangle = \{\mathbf{U}_i \in \mathcal{A} \mid \mathbf{U}_i \nsubseteq \mathbf{W}\}$.

Theorem: $\bigvee\{\Pi_{\mathbf{U}_i} \mid \mathbf{U}_i \in \text{JCompl}\langle \mathbf{W}, \mathcal{A}\rangle\}$ is an optimal $\vee\Pi$-complement of $\Pi_{\mathbf{W}}$. □

Issues with $\bigvee\Pi$-complements

- For a wide variety of constraints on the main schema, the constraints on a Π-view are well behaved first-order database dependencies [Fagin 82 JACM] [Hull 84 JACM].

Issues with $\bigvee\Pi$-complements

- For a wide variety of constraints on the main schema, the constraints on a Π-view are well behaved first-order database dependencies [Fagin 82 JACM] [Hull 84 JACM].
- For $\bigvee\Pi$-views, the situation is very different.

Issues with $\bigvee\Pi$-complements

- For a wide variety of constraints on the main schema, the constraints on a Π-view are well behaved first-order database dependencies [Fagin 82 JACM] [Hull 84 JACM].
- For $\bigvee\Pi$-views, the situation is very different.

Example context continued: $R[ABCDE]\ C \to D\ \ D \to E\ \models\ \bowtie[ABC, CD, DE]$.

Issues with $\bigvee\Pi$-complements

- For a wide variety of constraints on the main schema, the constraints on a Π-view are well behaved first-order database dependencies [Fagin 82 JACM] [Hull 84 JACM].
- For $\bigvee\Pi$-views, the situation is very different.

Example context continued: $R[ABCDE]\ C \to D\ \ D \to E\ \models\ \bowtie [ABC, CD, DE]$.

- On $\Pi_{ABC} \vee \Pi_{DE}$, the constraint Cardinality$(\Pi_D) \leqslant$ Cardinality(Π_C) holds.

Issues with $\bigvee\Pi$-complements

- For a wide variety of constraints on the main schema, the constraints on a Π-view are well behaved first-order database dependencies [Fagin 82 JACM] [Hull 84 JACM].

- For $\bigvee\Pi$-views, the situation is very different.

Example context continued: $R[ABCDE]\ C \to D \ \ D \to E \ \models \ \bowtie [ABC, CD, DE]$.

- On $\Pi_{ABC} \vee \Pi_{DE}$, the constraint Cardinality$(\Pi_D) \leqslant$ Cardinality(Π_C) holds.

- It is not even first order for infinite databases.

Issues with $\bigvee\Pi$-complements

- For a wide variety of constraints on the main schema, the constraints on a Π-view are well behaved first-order database dependencies [Fagin 82 JACM] [Hull 84 JACM].

- For $\bigvee\Pi$-views, the situation is very different.

Example context continued: $R[ABCDE]\ C \to D\ \ D \to E\ \models\ \bowtie[ABC, CD, DE]$.

- On $\Pi_{ABC} \vee \Pi_{DE}$, the constraint Cardinality$(\Pi_D) \leqslant$ Cardinality(Π_C) holds.

- It is not even first order for infinite databases.

- Fortunately, it does not matter.

Issues with $\bigvee\Pi$-complements

- For a wide variety of constraints on the main schema, the constraints on a Π-view are well behaved first-order database dependencies [Fagin 82 JACM] [Hull 84 JACM].
- For $\bigvee\Pi$-views, the situation is very different.

Example context continued: $R[ABCDE]\ C \to D\ \ D \to E\ \models\ \bowtie[ABC, CD, DE]$.

- On $\Pi_{ABC} \vee \Pi_{DE}$, the constraint Cardinality$(\Pi_D) \leqslant$ Cardinality(Π_C) holds.
- It is not even first order for infinite databases.
- Fortunately, it does not matter.
- The truth value of such constraints is never altered by a constant-complement update [Hegner 06 AMAI].

Issues with $\bigvee\Pi$-complements

- For a wide variety of constraints on the main schema, the constraints on a Π-view are well behaved first-order database dependencies [Fagin 82 JACM] [Hull 84 JACM].
- For $\bigvee\Pi$-views, the situation is very different.

Example context continued: $R[ABCDE]\ C \to D\ \ D \to E\ \models\ \bowtie[ABC, CD, DE]$.

- On $\Pi_{ABC} \vee \Pi_{DE}$, the constraint Cardinality$(\Pi_D) \leqslant$ Cardinality(Π_C) holds.
- It is not even first order for infinite databases.
- Fortunately, it does not matter.
- The truth value of such constraints is never altered by a constant-complement update [Hegner 06 AMAI].
- Only "simple" constraints must be checked for an update.

Goal: Carry the theory of optimal complements beyond projections.

Moving Beyond the Framework of Projections

Goal: Carry the theory of optimal complements beyond projections.

- At least include selections, and preferably joins.

Moving Beyond the Framework of Projections

Goal: Carry the theory of optimal complements beyond projections.

- At least include selections, and preferably joins.

Principles: Look for a more general theory, as opposed to an approach based upon individual cases.

Moving Beyond the Framework of Projections

Goal: Carry the theory of optimal complements beyond projections.

- At least include selections, and preferably joins.

Principles: Look for a more general theory, as opposed to an approach based upon individual cases.

General contexts:

Moving Beyond the Framework of Projections

Goal: Carry the theory of optimal complements beyond projections.

- At least include selections, and preferably joins.

Principles: Look for a more general theory, as opposed to an approach based upon individual cases.

General contexts:

- For general principles of schemata and views, for the definition of optimality: a simple set-based context.

Moving Beyond the Framework of Projections

Goal: Carry the theory of optimal complements beyond projections.

- At least include selections, and preferably joins.

Principles: Look for a more general theory, as opposed to an approach based upon individual cases.

General contexts:

- For general principles of schemata and views, for the definition of optimality: a simple set-based context.
- For the characterization of views, *information* based upon Boolean queries.

Moving Beyond the Framework of Projections

Goal: Carry the theory of optimal complements beyond projections.

- At least include selections, and preferably joins.

Principles: Look for a more general theory, as opposed to an approach based upon individual cases.

General contexts:

- For general principles of schemata and views, for the definition of optimality: a simple set-based context.
- For the characterization of views, *information* based upon Boolean queries.
- For decomposition, the *information semilattice* of equivalence classes of Boolean queries on the main schema.

Optimality in a General Context

- Comparison of views in a general setting is easy.

Optimality in a General Context

- Comparison of views in a general setting is easy.
- A database schema $\mathbf{D}$ has a set $\text{LDB}(\mathbf{D})$ of legal states.

Optimality in a General Context

- Comparison of views in a general setting is easy.
- A database schema $\mathbf{D}$ has a set $\mathsf{LDB}(\mathbf{D})$ of legal states.
- A view $\Gamma = (\mathbf{V}, \gamma)$ of $\mathbf{D}$ consists of a schema $\mathbf{V}$ together with a surjective morphism $\gamma : \mathsf{LDB}(\mathbf{D}) \to \mathsf{LDB}(\mathbf{V})$.

Optimality in a General Context

- Comparison of views in a general setting is easy.
- A database schema $\mathbf{D}$ has a set $\mathsf{LDB}(\mathbf{D})$ of legal states.
- A view $\Gamma = (\mathbf{V}, \gamma)$ of $\mathbf{D}$ consists of a schema $\mathbf{V}$ together with a surjective morphism $\gamma : \mathsf{LDB}(\mathbf{D}) \to \mathsf{LDB}(\mathbf{V})$.
- The *congruence* of $\Gamma = (\mathbf{V}, \gamma)$ is
$$\mathsf{Congr}(\Gamma) = \{(M_1, M_2) \in \mathsf{LDB}(\mathbf{D}) \times \mathsf{LDB}(\mathbf{D}) \mid \gamma(M_1) = \gamma(M_2)\}.$$

Optimality in a General Context

- Comparison of views in a general setting is easy.
- A database schema $\mathbf{D}$ has a set $\mathsf{LDB}(\mathbf{D})$ of legal states.
- A view $\Gamma = (\mathbf{V}, \gamma)$ of $\mathbf{D}$ consists of a schema $\mathbf{V}$
  together with a surjective morphism $\gamma : \mathsf{LDB}(\mathbf{D}) \to \mathsf{LDB}(\mathbf{V})$.
- The *congruence* of $\Gamma = (\mathbf{V}, \gamma)$ is
  $\mathsf{Congr}(\Gamma) = \{(M_1, M_2) \in \mathsf{LDB}(\mathbf{D}) \times \mathsf{LDB}(\mathbf{D}) \mid \gamma(M_1) = \gamma(M_2)\}$.
- Define $\Gamma_1 \preceq \Gamma_2$ iff $\mathsf{Congr}(\Gamma_2) \subseteq \mathsf{Congr}(\Gamma_1)$.

Optimality in a General Context

- Comparison of views in a general setting is easy.
- A database schema $\mathbf{D}$ has a set $\mathsf{LDB}(\mathbf{D})$ of legal states.
- A view $\Gamma = (\mathbf{V}, \gamma)$ of $\mathbf{D}$ consists of a schema $\mathbf{V}$ together with a surjective morphism $\gamma : \mathsf{LDB}(\mathbf{D}) \to \mathsf{LDB}(\mathbf{V})$.
- The *congruence* of $\Gamma = (\mathbf{V}, \gamma)$ is
$$\mathsf{Congr}(\Gamma) = \{(M_1, M_2) \in \mathsf{LDB}(\mathbf{D}) \times \mathsf{LDB}(\mathbf{D}) \mid \gamma(M_1) = \gamma(M_2)\}.$$
- Define $\Gamma_1 \preceq \Gamma_2$ iff $\mathsf{Congr}(\Gamma_2) \subseteq \mathsf{Congr}(\Gamma_1)$.
- This definition agrees with those given for Π-views and $\bigvee\Pi$-views.

Optimality in a General Context

- Comparison of views in a general setting is easy.
- A database schema $\mathbf{D}$ has a set $\mathsf{LDB}(\mathbf{D})$ of legal states.
- A view $\Gamma = (\mathbf{V}, \gamma)$ of $\mathbf{D}$ consists of a schema $\mathbf{V}$ together with a surjective morphism $\gamma : \mathsf{LDB}(\mathbf{D}) \to \mathsf{LDB}(\mathbf{V})$.
- The *congruence* of $\Gamma = (\mathbf{V}, \gamma)$ is $\mathsf{Congr}(\Gamma) = \{(M_1, M_2) \in \mathsf{LDB}(\mathbf{D}) \times \mathsf{LDB}(\mathbf{D}) \mid \gamma(M_1) = \gamma(M_2)\}$.
- Define $\Gamma_1 \preceq \Gamma_2$ iff $\mathsf{Congr}(\Gamma_2) \subseteq \mathsf{Congr}(\Gamma_1)$.
- This definition agrees with those given for Π-views and $\bigvee\Pi$-views.
- Thus, view Γ is optimal in a class $\mathcal{V}$ if its congruence is least over all elements of $\mathcal{V}$.

Optimality in a General Context

- Comparison of views in a general setting is easy.
- A database schema $\mathbf{D}$ has a set $\mathsf{LDB}(\mathbf{D})$ of legal states.
- A view $\Gamma = (\mathbf{V}, \gamma)$ of $\mathbf{D}$ consists of a schema $\mathbf{V}$ together with a surjective morphism $\gamma : \mathsf{LDB}(\mathbf{D}) \to \mathsf{LDB}(\mathbf{V})$.
- The *congruence* of $\Gamma = (\mathbf{V}, \gamma)$ is
$$\mathsf{Congr}(\Gamma) = \{(M_1, M_2) \in \mathsf{LDB}(\mathbf{D}) \times \mathsf{LDB}(\mathbf{D}) \mid \gamma(M_1) = \gamma(M_2)\}.$$
- Define $\Gamma_1 \preceq \Gamma_2$ iff $\mathsf{Congr}(\Gamma_2) \subseteq \mathsf{Congr}(\Gamma_1)$.
- This definition agrees with those given for Π-views and $\bigvee\Pi$-views.
- Thus, view Γ is optimal in a class $\mathcal{V}$ if its congruence is least over all elements of $\mathcal{V}$.
- Such a view is unique up to the isomorphism class defined by congruence.

Views Based upon Conjunctive Queries

- A *conjunctive query* on a relational schema is a formula defined using only $\wedge$ and $\exists$.

Views Based upon Conjunctive Queries

- A *conjunctive query* on a relational schema is a formula defined using only $\wedge$ and $\exists$.
- (Single-value) selection, projection, and join are defined by such queries using the relational algebra:

Views Based upon Conjunctive Queries

- A *conjunctive query* on a relational schema is a formula defined using only $\wedge$ and $\exists$.
- (Single-value) selection, projection, and join are defined by such queries using the relational algebra:

  Example: $\pi_{AB}(R[ABC])$ is defined by $(\exists z)(R(x_A, x_B, z))$.

Views Based upon Conjunctive Queries

- A *conjunctive query* on a relational schema is a formula defined using only $\wedge$ and $\exists$.
- (Single-value) selection, projection, and join are defined by such queries using the relational algebra:

  Example: $\pi_{AB}(R[ABC])$ is defined by $(\exists z)(R(x_A, x_B, z))$.

  Example: $\sigma_{A=\mathrm{a}}(R[ABC])$ is defined by $R(\mathrm{a}, x_B, x_C)$.

Views Based upon Conjunctive Queries

- A *conjunctive query* on a relational schema is a formula defined using only $\wedge$ and $\exists$.
- (Single-value) selection, projection, and join are defined by such queries using the relational algebra:

  Example: $\pi_{AB}(R[ABC])$ is defined by $(\exists z)(R(x_A, x_B, z))$.

  Example: $\sigma_{A=\mathrm{a}}(R[ABC])$ is defined by $R(\mathrm{a}, x_B, x_C)$.

- These define the $\exists\wedge+$-views.

Views Based upon Conjunctive Queries

- A *conjunctive query* on a relational schema is a formula defined using only $\wedge$ and $\exists$.
- (Single-value) selection, projection, and join are defined by such queries using the relational algebra:

  Example: $\pi_{AB}(R[ABC])$ is defined by $(\exists z)(R(x_A, x_B, z))$.

  Example: $\sigma_{A=\mathrm{a}}(R[ABC])$ is defined by $R(\mathrm{a}, x_B, x_C)$.

- These define the $\exists\wedge+$-views.
- A *Boolean conjunctive query* or $\exists\wedge+$-*query* contains no free variables.

Views Based upon Conjunctive Queries

- A *conjunctive query* on a relational schema is a formula defined using only $\wedge$ and $\exists$.
- (Single-value) selection, projection, and join are defined by such queries using the relational algebra:

  Example: $\pi_{AB}(R[ABC])$ is defined by $(\exists z)(R(x_A, x_B, z))$.

  Example: $\sigma_{A=\mathrm{a}}(R[ABC])$ is defined by $R(\mathrm{a}, x_B, x_C)$.

- These define the $\exists\wedge+$-views.
- A *Boolean conjunctive query* or $\exists\wedge+$*-query* contains no free variables.
- The tuples in $\exists\wedge+$-views correspond to Boolean conjunctive queries on the main schema

Views Based upon Conjunctive Queries

- A *conjunctive query* on a relational schema is a formula defined using only $\wedge$ and $\exists$.
- (Single-value) selection, projection, and join are defined by such queries using the relational algebra:

  Example: $\pi_{AB}(R[ABC])$ is defined by $(\exists z)(R(x_A, x_B, z))$.

  Example: $\sigma_{A=\mathrm{a}}(R[ABC])$ is defined by $R(\mathrm{a}, x_B, x_C)$.

- These define the $\exists\wedge+$-views.
- A *Boolean conjunctive query* or *$\exists\wedge+$-query* contains no free variables.
- The tuples in $\exists\wedge+$-views correspond to Boolean conjunctive queries on the main schema

  Example: The tuple (b, c) for the view defined by $\pi_{AB}(R[ABC])$ corresponds to the Boolean query $(\exists z)(R(\mathrm{a}, \mathrm{b}, z))$.

Views Based upon Conjunctive Queries

- A *conjunctive query* on a relational schema is a formula defined using only $\wedge$ and $\exists$.
- (Single-value) selection, projection, and join are defined by such queries using the relational algebra:

  Example: $\pi_{AB}(R[ABC])$ is defined by $(\exists z)(R(x_A, x_B, z))$.

  Example: $\sigma_{A=\mathrm{a}}(R[ABC])$ is defined by $R(\mathrm{a}, x_B, x_C)$.

- These define the $\exists\wedge+$-views.
- A *Boolean conjunctive query* or *$\exists\wedge+$-query* contains no free variables.
- The tuples in $\exists\wedge+$-views correspond to Boolean conjunctive queries on the main schema

  Example: The tuple (b, c) for the view defined by $\pi_{AB}(R[ABC])$ corresponds to the Boolean query $(\exists z)(R(\mathrm{a}, \mathrm{b}, z))$.

  Example: The tuple $(\mathrm{a}, \mathrm{b}, \mathrm{c})$ for the view defined by $\sigma_{A=\mathrm{a}}(R[ABC])$ corresponds to the Boolean query $R(\mathrm{a}, \mathrm{b}, \mathrm{c})$.

Extending and Limiting Views Based upon Conjunctive Queries

Extensions:

- A limitation of $\exists\wedge+$-views is that they recapture only single-valued selection.

Extensions:

- A limitation of $\exists\wedge+$-views is that they recapture only single-valued selection.

Example: A selection such as $\sigma_{(A\leqslant 30)}(R[ABC])$ is not recaptured.

Extending and Limiting Views Based upon Conjunctive Queries

Extensions:

- A limitation of $\exists\wedge+$-views is that they recapture only single-valued selection.

Example: A selection such as $\sigma_{(A \leqslant 30)}(R[ABC])$ is not recaptured.

- The developed framework supports such $\sigma\exists\wedge+$-queries for defining views.

Extending and Limiting Views Based upon Conjunctive Queries

Extensions:

- A limitation of $\exists\wedge+$-views is that they recapture only single-valued selection.

Example: A selection such as $\sigma_{(A\leqslant 30)}(R[ABC])$ is not recaptured.

- The developed framework supports such $\sigma\exists\wedge+$-queries for defining views.
- Any subset selection is allowed.

Extending and Limiting Views Based upon Conjunctive Queries

Extensions:

- A limitation of $\exists\wedge+$-views is that they recapture only single-valued selection.

Example: A selection such as $\sigma_{(A\leqslant 30)}(R[ABC])$ is not recaptured.

- The developed framework supports such $\sigma\exists\wedge+$-queries for defining views.
- Any subset selection is allowed.

Limitations:

- For technical reasons, view definitions which "hide" constants are not allowed.

Extensions:

- A limitation of $\exists\wedge+$-views is that they recapture only single-valued selection.

Example: A selection such as $\sigma_{(A\leqslant 30)}(R[ABC])$ is not recaptured.

- The developed framework supports such $\sigma\exists\wedge+$-queries for defining views.
- Any subset selection is allowed.

- For technical reasons, view definitions which "hide" constants are not allowed.

  Example: The two definitions $\pi_{BC}(\sigma_{A=\mathrm{a}_1}(R[ABC])$ and $\pi_{BC}(\sigma_{A=\mathrm{a}_2}(R[ABC])$; hide their selection constant in the sense that it is not visible in the view.

The Representation of $\sigma\exists\wedge+$-Views using sets of Boolean Queries

- A relation R in a $\sigma\exists\wedge+$-view Γ may be represented by the set DisjRep$\langle\Gamma, R\rangle$ of all $\exists\wedge+$-queries which are obtained by grounding its defining formula.

The Representation of $\sigma\exists\wedge+$-Views using sets of Boolean Queries

- A relation R in a $\sigma\exists\wedge+$-view Γ may be represented by the set $\mathsf{DisjRep}\langle\Gamma, R\rangle$ of all $\exists\wedge+$-queries which are obtained by grounding its defining formula.
- In this approach, each Boolean query corresponds to a possible view tuple.

The Representation of $\sigma\exists\wedge+$-Views using sets of Boolean Queries

- A relation R in a $\sigma\exists\wedge+$-view Γ may be represented by the set DisjRep$\langle\Gamma, R\rangle$ of all $\exists\wedge+$-queries which are obtained by grounding its defining formula.
- In this approach, each Boolean query corresponds to a possible view tuple.

Example: The view defined by $\pi_{AB}(R[ABC])$ corresponds to the set $\{(\exists z)(R(a, b, z)) \mid a, b, \in \text{Const}\}$.

The Representation of $\sigma\exists\wedge+$-Views using sets of Boolean Queries

- A relation R in a $\sigma\exists\wedge+$-view Γ may be represented by the set DisjRep$\langle\Gamma, R\rangle$ of all $\exists\wedge+$-queries which are obtained by grounding its defining formula.
- In this approach, each Boolean query corresponds to a possible view tuple.

Example: The view defined by $\pi_{AB}(R[ABC])$ corresponds to the set
$\{(\exists z)(R(a, b, z)) \mid a, b, \in \text{Const}\}$.

Example: The view defined by $\sigma_{A=\mathrm{a}}(R[ABC])$ corresponds to the set
$\{(R(\mathrm{a}, b, c)) \mid b, c \in \text{Const}\}$.

The Representation of $\sigma\exists\wedge+$-Views using sets of Boolean Queries

- A relation R in a $\sigma\exists\wedge+$-view Γ may be represented by the set $\mathsf{DisjRep}\langle\Gamma, R\rangle$ of all $\exists\wedge+$-queries which are obtained by grounding its defining formula.
- In this approach, each Boolean query corresponds to a possible view tuple.

Example: The view defined by $\pi_{AB}(R[ABC])$ corresponds to the set
$$\{(\exists z)(R(a,b,z)) \mid a,b, \in \mathsf{Const}\}.$$

Example: The view defined by $\sigma_{A=\mathrm{a}}(R[ABC])$ corresponds to the set
$$\{(R(\mathrm{a},b,c)) \mid b,c \in \mathsf{Const}\}.$$

Example: The view defined by $\sigma_{A\in S}(R[ABC])$ corresponds to the set
$$\{(R(a,b,c)) \mid (a \in S) \wedge (b,c \in \mathsf{Const})\}.$$

The Representation of $\sigma\exists\wedge+$-Views using sets of Boolean Queries

- A relation R in a $\sigma\exists\wedge+$-view Γ may be represented by the set $\mathsf{DisjRep}\langle\Gamma, R\rangle$ of all $\exists\wedge+$-queries which are obtained by grounding its defining formula.
- In this approach, each Boolean query corresponds to a possible view tuple.

  Example: The view defined by $\pi_{AB}(R[ABC])$ corresponds to the set
  $$\{(\exists z)(R(a, b, z)) \mid a, b, \in \mathsf{Const}\}.$$
  Example: The view defined by $\sigma_{A=\mathrm{a}}(R[ABC])$ corresponds to the set
  $$\{(R(\mathrm{a}, b, c)) \mid b, c \in \mathsf{Const}\}.$$
  Example: The view defined by $\sigma_{A\in S}(R[ABC])$ corresponds to the set
  $$\{(R(a, b, c)) \mid (a \in S) \wedge (b, c \in \mathsf{Const})\}.$$
- The goal is to be able to represent all relations in the view using a single set of Boolean queries.

The Representation of $\sigma\exists\wedge+$-Views using sets of Boolean Queries

- A relation R in a $\sigma\exists\wedge+$-view Γ may be represented by the set $\text{DisjRep}\langle\Gamma, R\rangle$ of all $\exists\wedge+$-queries which are obtained by grounding its defining formula.
- In this approach, each Boolean query corresponds to a possible view tuple.

  Example: The view defined by $\pi_{AB}(R[ABC])$ corresponds to the set
  $$\{(\exists z)(R(a, b, z)) \mid a, b, \in \text{Const}\}.$$
  Example: The view defined by $\sigma_{A=\mathrm{a}}(R[ABC])$ corresponds to the set
  $$\{(R(\mathrm{a}, b, c)) \mid b, c \in \text{Const}\}.$$
  Example: The view defined by $\sigma_{A\in S}(R[ABC])$ corresponds to the set
  $$\{(R(a, b, c)) \mid (a \in S) \wedge (b, c \in \text{Const})\}.$$
- The goal is to be able to represent all relations in the view using a single set of Boolean queries.
- This means that the view relation must be recoverable from information in the Boolean query.

The Representation of $\sigma\exists\wedge+$-Views using sets of Boolean Queries

- A relation R in a $\sigma\exists\wedge+$-view Γ may be represented by the set DisjRep$\langle\Gamma, R\rangle$ of all $\exists\wedge+$-queries which are obtained by grounding its defining formula.
- In this approach, each Boolean query corresponds to a possible view tuple.

  Example: The view defined by $\pi_{AB}(R[ABC])$ corresponds to the set
  $$\{(\exists z)(R(a, b, z)) \mid a, b, \in \text{Const}\}.$$
  Example: The view defined by $\sigma_{A=\mathrm{a}}(R[ABC])$ corresponds to the set
  $$\{(R(\mathrm{a}, b, c)) \mid b, c \in \text{Const}\}.$$
  Example: The view defined by $\sigma_{A\in S}(R[ABC])$ corresponds to the set
  $$\{(R(a, b, c)) \mid (a \in S) \wedge (b, c \in \text{Const})\}.$$

- The goal is to be able to represent all relations in the view using a single set of Boolean queries.
- This means that the view relation must be recoverable from information in the Boolean query.
- The $\exists\wedge+$-formula defining the view is called its *pattern*.

The Representation of $\sigma\exists\wedge+$-Views using sets of Boolean Queries

- A relation R in a $\sigma\exists\wedge+$-view Γ may be represented by the set DisjRep$\langle\Gamma, R\rangle$ of all $\exists\wedge+$-queries which are obtained by grounding its defining formula.
- In this approach, each Boolean query corresponds to a possible view tuple.

  Example: The view defined by $\pi_{AB}(R[ABC])$ corresponds to the set
  $\{(\exists z)(R(a, b, z)) \mid a, b, \in \text{Const}\}$.

  Example: The view defined by $\sigma_{A=\mathrm{a}}(R[ABC])$ corresponds to the set
  $\{(R(\mathrm{a}, b, c)) \mid b, c \in \text{Const}\}$.

  Example: The view defined by $\sigma_{A\in S}(R[ABC])$ corresponds to the set
  $\{(R(a, b, c)) \mid (a \in S) \wedge (b, c \in \text{Const})\}$.

- The goal is to be able to represent all relations in the view using a single set of Boolean queries.
- This means that the view relation must be recoverable from information in the Boolean query.
- The $\exists\wedge+$-formula defining the view is called its *pattern*.
- Each Boolean query must correspond to a single pattern.

Concrete and Abstract Views

- A *concrete* view is defined in the usual way, using the relational calculus restricted to the $\sigma\exists\wedge+$-context.

Concrete and Abstract Views

- A *concrete* view is defined in the usual way, using the relational calculus restricted to the $\sigma\exists\wedge+$-context.
- An *abstract view* consists of a set of Boolean queries, subject to the constraint that it is of *finite pattern index*.

Concrete and Abstract Views

- A *concrete* view is defined in the usual way, using the relational calculus restricted to the $\sigma\exists\wedge+$-context.
- An *abstract view* consists of a set of Boolean queries, subject to the constraint that it is of *finite pattern index*.
  - This means that there is a finite set of patterns, and each of the queries matches one of those patterns.

Concrete and Abstract Views

- A *concrete* view is defined in the usual way, using the relational calculus restricted to the $\sigma\exists\wedge+$-context.
- An *abstract view* consists of a set of Boolean queries, subject to the constraint that it is of *finite pattern index*.
  - This means that there is a finite set of patterns, and each of the queries matches one of those patterns.
  - This property is essential for recovering a concrete view from an abstract one.

Concrete and Abstract Views

- A *concrete* view is defined in the usual way, using the relational calculus restricted to the $\sigma\exists\wedge+$-context.
- An *abstract view* consists of a set of Boolean queries, subject to the constraint that it is of *finite pattern index*.
  - This means that there is a finite set of patterns, and each of the queries matches one of those patterns.
  - This property is essential for recovering a concrete view from an abstract one.

Theorem; There is a natural correspondence between concrete and abstract views. □

The Information Semilattice and the Decomposition Basis

- Write $\varphi_1 \equiv_{\mathbf{D}} \varphi_2$ if the two Boolean queries have the same truth value on every $M \in \mathsf{LDB}(\mathbf{D})$.

The Information Semilattice and the Decomposition Basis

- Write $\varphi_1 \equiv_{\mathbf{D}} \varphi_2$ if the two Boolean queries have the same truth value on every $M \in \mathsf{LDB}(\mathbf{D})$.

  Example: $R(\mathrm{a}, \mathrm{b}, \mathrm{c}) \equiv_{\mathbf{D}} (\exists z)(R(\mathrm{a}, \mathrm{b}, z) \wedge (\exists x)(R(x, \mathrm{b}, \mathrm{c}))$ if the JD $\bowtie [AB, BC]$ holds.

- Write $[\varphi]$ for the induced equivalence class on φ.

The Information Semilattice and the Decomposition Basis

- Write $\varphi_1 \equiv_{\mathbf{D}} \varphi_2$ if the two Boolean queries have the same truth value on every $M \in \text{LDB}(\mathbf{D})$.

  Example: $R(\text{a}, \text{b}, \text{c}) \equiv_{\mathbf{D}} (\exists z)(R(\text{a}, \text{b}, z) \wedge (\exists x)(R(x, \text{b}, \text{c}))$ if the JD $\bowtie [AB, BC]$ holds.

- Write $[\varphi]$ for the induced equivalence class on φ.
- Write $[\varphi_1] \sqsubseteq_{\equiv_{\mathbf{D}}} [\varphi_2]$ if $[\varphi_2]$ is true on $\mathbf{D}$ whenever $[\varphi_1]$ is.

The Information Semilattice and the Decomposition Basis

- Write $\varphi_1 \equiv_{\mathbf{D}} \varphi_2$ if the two Boolean queries have the same truth value on every $M \in \mathsf{LDB}(\mathbf{D})$.

  Example: $R(\mathrm{a}, \mathrm{b}, \mathrm{c}) \equiv_{\mathbf{D}} (\exists z)(R(\mathrm{a}, \mathrm{b}, z) \wedge (\exists x)(R(x, \mathrm{b}, \mathrm{c}))$ if the JD $\bowtie [AB, BC]$ holds.

- Write $[\varphi]$ for the induced equivalence class on φ.
- Write $[\varphi_1] \sqsubseteq_{\equiv_{\mathbf{D}}} [\varphi_2]$ if $[\varphi_2]$ is true on $\mathbf{D}$ whenever $[\varphi_1]$ is.
- This set forms a meet semilattice with top element $[\mathbf{false}]$ and bottom element $[\mathbf{true}]$.

The Information Semilattice and the Decomposition Basis

- Write $\varphi_1 \equiv_{\mathbf{D}} \varphi_2$ if the two Boolean queries have the same truth value on every $M \in \mathsf{LDB}(\mathbf{D})$.

  Example: $R(\mathrm{a}, \mathrm{b}, \mathrm{c}) \equiv_{\mathbf{D}} (\exists z)(R(\mathrm{a}, \mathrm{b}, z) \wedge (\exists x)(R(x, \mathrm{b}, \mathrm{c}))$ if the JD $\bowtie [AB, BC]$ holds.

- Write $[\varphi]$ for the induced equivalence class on φ.
- Write $[\varphi_1] \sqsubseteq_{\equiv_{\mathbf{D}}} [\varphi_2]$ if $[\varphi_2]$ is true on $\mathbf{D}$ whenever $[\varphi_1]$ is.
- This set forms a meet semilattice with top element $[\mathbf{false}]$ and bottom element $[\mathbf{true}]$.
- The key idea is to look for a *decomposition* basis in this semilattice. Roughly, a sentence is in the decomposition basis if

The Information Semilattice and the Decomposition Basis

- Write $\varphi_1 \equiv_{\mathbf{D}} \varphi_2$ if the two Boolean queries have the same truth value on every $M \in \mathsf{LDB}(\mathbf{D})$.

  Example: $R(\mathrm{a}, \mathrm{b}, \mathrm{c}) \equiv_{\mathbf{D}} (\exists z)(R(\mathrm{a}, \mathrm{b}, z) \wedge (\exists x)(R(x, \mathrm{b}, \mathrm{c}))$ if the JD $\bowtie [AB, BC]$ holds.

- Write $[\varphi]$ for the induced equivalence class on φ.
- Write $[\varphi_1] \sqsubseteq_{\equiv_{\mathbf{D}}} [\varphi_2]$ if $[\varphi_2]$ is true on $\mathbf{D}$ whenever $[\varphi_1]$ is.
- This set forms a meet semilattice with top element $[\mathbf{false}]$ and bottom element $[\mathbf{true}]$.
- The key idea is to look for a *decomposition* basis in this semilattice. Roughly, a sentence is in the decomposition basis if
  - it is a useful in a nontrivial way in the representation of a tuple as a join, and

The Information Semilattice and the Decomposition Basis

- Write $\varphi_1 \equiv_{\mathbf{D}} \varphi_2$ if the two Boolean queries have the same truth value on every $M \in \mathsf{LDB}(\mathbf{D})$.

  Example: $R(\mathrm{a}, \mathrm{b}, \mathrm{c}) \equiv_{\mathbf{D}} (\exists z)(R(\mathrm{a}, \mathrm{b}, z) \wedge (\exists x)(R(x, \mathrm{b}, \mathrm{c}))$ if the JD $\bowtie [AB, BC]$ holds.

- Write $[\varphi]$ for the induced equivalence class on φ.
- Write $[\varphi_1] \sqsubseteq_{\equiv_{\mathbf{D}}} [\varphi_2]$ if $[\varphi_2]$ is true on $\mathbf{D}$ whenever $[\varphi_1]$ is.
- This set forms a meet semilattice with top element $[\mathbf{false}]$ and bottom element $[\mathbf{true}]$.
- The key idea is to look for a *decomposition* basis in this semilattice. Roughly, a sentence is in the decomposition basis if
  - it is a useful in a nontrivial way in the representation of a tuple as a join, and
  - it cannot be further decomposed in a nontrivial way.

The Information Semilattice and the Decomposition Basis

- Write $\varphi_1 \equiv_{\mathbf{D}} \varphi_2$ if the two Boolean queries have the same truth value on every $M \in \mathsf{LDB}(\mathbf{D})$.

  Example: $R(\mathrm{a}, \mathrm{b}, \mathrm{c}) \equiv_{\mathbf{D}} (\exists z)(R(\mathrm{a}, \mathrm{b}, z) \wedge (\exists x)(R(x, \mathrm{b}, \mathrm{c}))$ if the JD $\bowtie [AB, BC]$ holds.

- Write $[\varphi]$ for the induced equivalence class on φ.
- Write $[\varphi_1] \sqsubseteq_{\equiv_{\mathbf{D}}} [\varphi_2]$ if $[\varphi_2]$ is true on $\mathbf{D}$ whenever $[\varphi_1]$ is.
- This set forms a meet semilattice with top element $[\mathbf{false}]$ and bottom element $[\mathbf{true}]$.
- The key idea is to look for a *decomposition* basis in this semilattice. Roughly, a sentence is in the decomposition basis if
  - it is a useful in a nontrivial way in the representation of a tuple as a join, and
  - it cannot be further decomposed in a nontrivial way.

Example: For the schema $R[ABC]$ constrained by $\bowtie [AB, BC]$, the decomposition basis consists of elements of the form $(\exists z)(R(\mathrm{a}, \mathrm{b}, z))$ and $(\exists x)(R(x, \mathrm{b}, \mathrm{c}))$.

Optimal Complements in a General Setting

Let $\Gamma = (\mathbf{V}, \gamma)$ be the view whose optimal complement is to be determined.

- All elements of the decomposition basis which "fit" into Γ are "placed" there.

Optimal Complements in a General Setting

Let $\Gamma = (\mathbf{V}, \gamma)$ be the view whose optimal complement is to be determined.

- All elements of the decomposition basis which "fit" into Γ are "placed" there.
    - In the case of a JD, this corresponds to identifying those projections which are subsumed by some projection of the JD.

Optimal Complements in a General Setting

Let $\Gamma = (\mathbf{V}, \gamma)$ be the view whose optimal complement is to be determined.

- All elements of the decomposition basis which "fit" into Γ are "placed" there.
  - In the case of a JD, this corresponds to identifying those projections which are subsumed by some projection of the JD.
- All other elements of the decomposition basis are used to generate a complement.
  - In the case of a JD, this corresponds to generating a complement from all projections of the JD which are not subsumed by the view to be complemented.

Optimal Complements in a General Setting

Let $\Gamma = (\mathbf{V}, \gamma)$ be the view whose optimal complement is to be determined.

- All elements of the decomposition basis which "fit" into Γ are "placed" there.
  - In the case of a JD, this corresponds to identifying those projections which are subsumed by some projection of the JD.
- All other elements of the decomposition basis are used to generate a complement.
  - In the case of a JD, this corresponds to generating a complement from all projections of the JD which are not subsumed by the view to be complemented.
- The condition for optimality of a complement is that upon ultimate decompositions of tuples using the decomposition basis are unique.

Optimal Complements in a General Setting

Let $\Gamma = (\mathbf{V}, \gamma)$ be the view whose optimal complement is to be determined.

- All elements of the decomposition basis which "fit" into Γ are "placed" there.
  - In the case of a JD, this corresponds to identifying those projections which are subsumed by some projection of the JD.
- All other elements of the decomposition basis are used to generate a complement.
  - In the case of a JD, this corresponds to generating a complement from all projections of the JD which are not subsumed by the view to be complemented.
- The condition for optimality of a complement is that upon ultimate decompositions of tuples using the decomposition basis are unique.
  - In the context of a single JD, this reduces exactly to that JD being nonredundant.

Optimal Complements in a General Setting

Let $\Gamma = (\mathbf{V}, \gamma)$ be the view whose optimal complement is to be determined.

- All elements of the decomposition basis which "fit" into Γ are "placed" there.
  - In the case of a JD, this corresponds to identifying those projections which are subsumed by some projection of the JD.
- All other elements of the decomposition basis are used to generate a complement.
  - In the case of a JD, this corresponds to generating a complement from all projections of the JD which are not subsumed by the view to be complemented.
- The condition for optimality of a complement is that upon ultimate decompositions of tuples using the decomposition basis are unique.
  - In the context of a single JD, this reduces exactly to that JD being nonredundant.
- There are of course many details which have been omitted.

Conclusions

- A study of the notion of optimal complements for views of relational schemata has been initiated.

Conclusions

- A study of the notion of optimal complements for views of relational schemata has been initiated.
- A simple and concrete representation for complements of views defined via projections has been developed fully.

Conclusions

- A study of the notion of optimal complements for views of relational schemata has been initiated.
- A simple and concrete representation for complements of views defined via projections has been developed fully.
- If the governing JD is dependency preserving, then this representation furthermore produces meet complements and so is appropriate for the constant-complement update strategy.

Conclusions

- A study of the notion of optimal complements for views of relational schemata has been initiated.
- A simple and concrete representation for complements of views defined via projections has been developed fully.
- If the governing JD is dependency preserving, then this representation furthermore produces meet complements and so is appropriate for the constant-complement update strategy.
- It identifies in particular the situations in which all of the updates on a view which are supportable via constant-complement are supportable via a single complement, and hence via a single update strategy.

Conclusions

- A study of the notion of optimal complements for views of relational schemata has been initiated.
- A simple and concrete representation for complements of views defined via projections has been developed fully.
- If the governing JD is dependency preserving, then this representation furthermore produces meet complements and so is appropriate for the constant-complement update strategy.
- It identifies in particular the situations in which all of the updates on a view which are supportable via constant-complement are supportable via a single complement, and hence via a single update strategy.
- A more general theory, not restricted to projections but rather based upon information and Boolean queries has also been developed.

Conclusions

- A study of the notion of optimal complements for views of relational schemata has been initiated.
- A simple and concrete representation for complements of views defined via projections has been developed fully.
- If the governing JD is dependency preserving, then this representation furthermore produces meet complements and so is appropriate for the constant-complement update strategy.
- It identifies in particular the situations in which all of the updates on a view which are supportable via constant-complement are supportable via a single complement, and hence via a single update strategy.
- A more general theory, not restricted to projections but rather based upon information and Boolean queries has also been developed.
- That theory provides a beginning to a more general theory but leaves several further directions.

Further Directions

Effective identification of meet complements:

Further Directions

Effective identification of meet complements:

- The constant-complement update strategy requires meet complements.

Effective identification of meet complements:

- The constant-complement update strategy requires meet complements.
- This is equivalent to the dependency preservation of the decomposition.

Further Directions

Effective identification of meet complements:

- The constant-complement update strategy requires meet complements.
- This is equivalent to the dependency preservation of the decomposition.
- An effective means to check this property for more general classes of constraints and views is needed.

Further Directions

Effective identification of meet complements:

- The constant-complement update strategy requires meet complements.
- This is equivalent to the dependency preservation of the decomposition.
- An effective means to check this property for more general classes of constraints and views is needed.

Explicit development of a theory of decomposition which includes selection:

Effective identification of meet complements:

- The constant-complement update strategy requires meet complements.
- This is equivalent to the dependency preservation of the decomposition.
- An effective means to check this property for more general classes of constraints and views is needed.

Explicit development of a theory of decomposition which includes selection:

- A simple theory for $\bigvee\Pi$-views has been developed.

Further Directions

Effective identification of meet complements:

- The constant-complement update strategy requires meet complements.
- This is equivalent to the dependency preservation of the decomposition.
- An effective means to check this property for more general classes of constraints and views is needed.

Explicit development of a theory of decomposition which includes selection:

- A simple theory for $\bigvee\Pi$-views has been developed.
- It rests upon well-known results for Π-views and basic dependencies.

Further Directions

Effective identification of meet complements:

- The constant-complement update strategy requires meet complements.
- This is equivalent to the dependency preservation of the decomposition.
- An effective means to check this property for more general classes of constraints and views is needed.

Explicit development of a theory of decomposition which includes selection:

- A simple theory for $\bigvee\Pi$-views has been developed.
- It rests upon well-known results for Π-views and basic dependencies.
- An expanded framework which includes views defined by selection is suggested.

Further Directions

Effective identification of meet complements:

- The constant-complement update strategy requires meet complements.
- This is equivalent to the dependency preservation of the decomposition.
- An effective means to check this property for more general classes of constraints and views is needed.

Explicit development of a theory of decomposition which includes selection:

- A simple theory for $\bigvee\Pi$-views has been developed.
- It rests upon well-known results for Π-views and basic dependencies.
- An expanded framework which includes views defined by selection is suggested.