Null Values

Used to represent missing information

At least two kinds

- dne — does not exist
- unk — value exists but unknown

PILOT	FLIGHT	DATE	TIME
Cushing	615	5Oct	**unk**
dne	704	6Oct	5:50a
Cook	**dne**	**dne**	**dne**

"Does Not Exist" Null

Like adding an additional value to domain of an attribute

So it enlarges the set of possible relation instances

Issue: dne = dne?

Maybe no: LD → P

PILOT	FLIGHT	DATE	TIME
Cook	**dne**	**dne**	**dne**
Cadiz	**dne**	**dne**	**dne**

Maybe yes: FIRST MI LAST → AGE

FIRST	MI	LAST	AGE
Alex	**dne**	Cook	37
Alex	**dne**	Cook	39

Unknown Null

More like variables

Can view an instance with **unk** nulls as a set of possible fully-defined instances

PILOT	FLIGHT	DATE	TIME
Cushing	615	5Oct	**unk**
Cook	704	6Oct	5:50a
Cushing	**unk**	6Oct	**unk**

Some Possible Instances

PILOT	FLIGHT	DATE	TIME
Cushing	615	5Oct	5:00p
Cook	704	6Oct	5:50a
Cushing	872	6Oct	11:00p

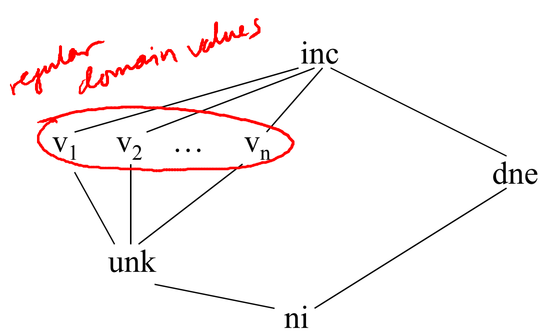
PILOT	FLIGHT	DATE	TIME
Cushing	615	5Oct	6:00p
Cook	704	6Oct	5:50a
Cushing	615	6Oct	6:00p

Also, “No-Information” and “Inconsistent”

$x \sqsubseteq y$: y more informative than x

ni $\sqsubseteq$ unk $\sqsubseteq$ v_i $\sqsubseteq$ inc

ni $\sqsubseteq$ dne $\sqsubseteq$ inc



Extend Order to Tuples

Use ⊑ on each component — partial order: reflexive, transitive, antisymmetric $t \sqsubseteq s \Rightarrow s \not\sqsubseteq t$

<Fred, unk, 6> ⊑ <Fred, 10.25, 6>
<Fred, unk, 6> ⊑ <Fred, unk, 6>
<ni, dne, 6> ⊑ <unk, inc, 6>
<ni, dne, 6> ⊑ <inc, inc, inc>

$\langle ni, dne, 6\rangle \not\sqsubseteq, \not\sqsupseteq \langle dne, ni, 6\rangle$

$\langle Fred, unk, 6\rangle \not\sqsubseteq, \not\sqsupseteq \langle Fran, unk, 6\rangle$

Extending to Relations

Concentrate on unk null

ord1(Person Price Amt)

Person	Price	Amt
Fred	unk	6
Fred	10.25	unk
Fritz	unk	unk

ord2(Person Price Amt)

Person	Price	Amt
Fred	11.50	6
unk	10.25	7

ord3(Person Price Amt)

Person	Price	Amt
Fred	11.50	6
Fred	10.25	7
Fritz	12.25	7

What Does a Tuple Mean

with nulls

For partial relations r, q, what should $r \sqsubseteq q$ require?

For tuple t in r
- exactly one s in q with $t \sqsubseteq s$?
- at least one s in q with $t \sqsubseteq s$?

For tuples t1, t2 in r, can there be a single s in q with $t1 \sqsubseteq s$ and $t2 \sqsubseteq s$?

Can there be s in q where $t \not\sqsubseteq s$ for all t in r? (closed world assumption)

Book's Definition

"Fill in the blanks"

every tuple in q gets mapped to

$r \sqsubseteq q$ means there is a total, onto map θ from r to q where for every tuple t in r

$$t \sqsubseteq \theta(t)$$

Examples

ord1 ⊑ ord3

```
ord1(Person Price Amt)  ord3(Person Price Amt)
     Fred   unk   6     ---> Fred   11.50 6
     Fred   10.25 unk   ---> Fred   10.25 7
     Fritz  unk   unk   ---> Fritz  12.25 7
```

θ

ord1 ⊑ ord4

```
ord1(Person Price Amt)  ord4(Person Price Amt)
     Fred   unk   6     ---> Fred   10.25 6
     Fred   10.25 unk   ---> Fritz  12.25 unk
     Fritz  unk   unk   --/
```

θ

But Consider

ord2 ⋢ ord3 (ord3 ⋢ ord2, also)

```
ord2(Person Price Amt)  ord3(Person Price Amt)
     Fred   11.50 6     ---> Fred   11.50 6
     unk    10.25 7     ---> Fred   10.25 7
                  ?     ---> Fritz  12.25 7
```

θ

unk1 unk2 7
unk3 unk4 7

Possible Worlds

View a partial relation as specifying a set of possible worlds

POSS(r) = {q | r ⊑ q and q is fully defined}

r1 ⊆ r2   POSS(r1) ⊇ POSS(r2)

Can include constraints C

POSS(r, C) = {q | q in POSS(r) and q in SAT(C)}

What Happens if We Have Constraints?

Recall

PILOT	FLIGHT	DATE	TIME
Cushing	615	5Oct	**unk**
Cook	704	6Oct	5:50a
Cushing	**unk**	6Oct	**unk**

Not All Satisfy the FDs

F = {L → T, PDT → L, LD → P}

PILOT	FLIGHT	DATE	TIME
Cushing	615	5Oct	5:00p
Cook	704	6Oct	5:50a
Cushing	704	6Oct	5:50a

LD ↛ P

PILOT	FLIGHT	DATE	TIME
Cushing	615	5Oct	6:00p
Cook	704	6Oct	5:50a
Cushing	615	6Oct	5:00p

L ↛ T

Use FD-Chase on Instances with Unknowns

r

PILOT	FLIGHT	DATE	TIME
Cushing	704	5Oct	**unk**
Cook	704	6Oct	5:50a
Cushing	**unk**	6Oct	**unk**

r′

PILOT	FLIGHT	DATE	TIME
Cushing	704	5Oct	5:50a
Cook	704	6Oct	5:50a
Cushing	**unk**	6Oct	**unk**

POSS(r, F) = POSS(r′, F)

Querying Relations with Unknowns

Let r be a relation with unknowns, and Q a query on full relations

Would like Q' on partial relations
- Q'(r) represents {Q(q) | q in POSS(r)}

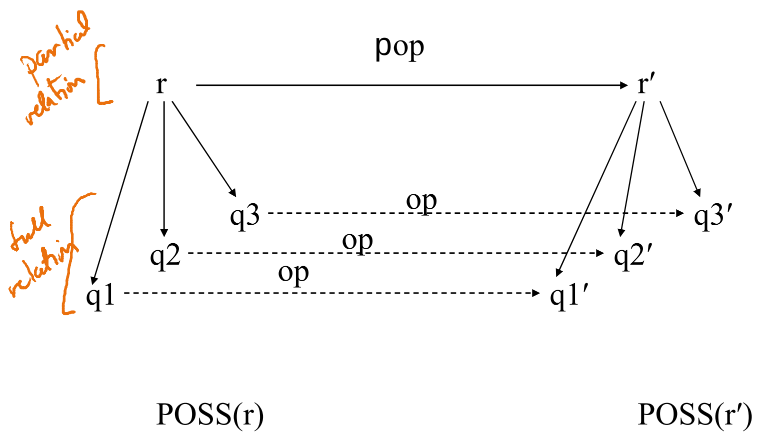
When r is fully defined, would like Q'(r) and Q(r) to agree. (Faithful)

Relational Algebra

One approach: Multi-relation
- Let pop be partial-relation version of operator op
- pπ and π

POSS(pop(r)) = {op(q) | q in POSS(r)}

Diagram



Sometimes Exists

r

PILOT	FLIGHT	DATE	TIME
Cushing	615	5Oct	**unk**
Cook	704	6Oct	5:50a
Cushing	**unk**	6Oct	**unk**

$p\pi_{DT}(r)$

DATE	TIME
5Oct	**unk**
6Oct	5:50a
6Oct	**unk**

need to treat unknowns as distinct

Not Always

no po in general

$\sigma_{L=872}(r)$

$r \sqsubseteq r1$ one way to fill in r

PILOT	FLIGHT	DATE	TIME
Cushing	615	5Oct	5:00p
Cook	704	6Oct	5:50a
Cushing	872	6Oct	11:00p

ans(r1)

PILOT	FLIGHT	DATE	TIME
Cushing	872	6Oct	11:00p

Principles of Databases

Not Always

$\sigma_{L=872}(r)$

$r \sqsubseteq r2$ a different way to fill in r

PILOT	FLIGHT	DATE	TIME
Cushing	615	5Oct	5:00p
Cook	704	6Oct	5:50a
Cushing	615	6Oct	5:00p

ans(r2)

PILOT	FLIGHT	DATE	TIME
∅			

no r' such that { ans(r1) ∈ POSS(r') and ans(r2) ∈ POSS(r') }

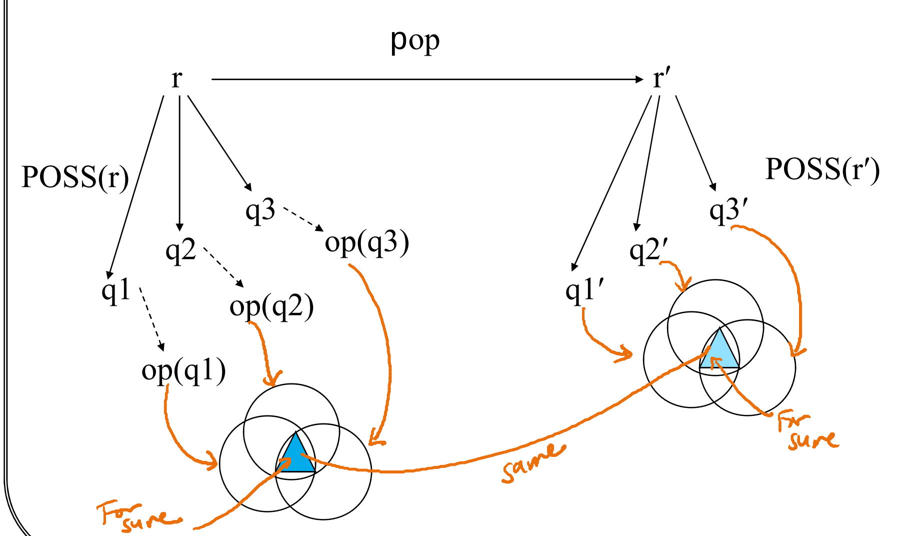
"For-sure semantics"

Truth preserving: common part of all possible results

$\cap$\{q | q in POSS(pop(r))\} =
$\cap$\{op(q')| q' in POSS(r)\}

all tuples that are in every possible answer
i.e. the "for-sure" tuples

Diagram



Projection

$p\pi_Y(r) = \{t[Y] \mid t \text{ in } r\}$

ord1(Person Price Amt)

Person	Price	Amt
Fred	unk	6
Fred	10.25	unk
Fritz	unk	unk

$p\pi_{Person\ Price}(ord1) =$

ans(Person Price)

Person	Price
Fred	unk
Fred	10.25
Fritz	unk

Projection 2

POSS(ord1) includes

(Person Price Amt)

Person	Price	Amt
Fred	11.50	6
Fred	10.25	7
Fritz	12.25	7

POSS($p\pi_{Person\ Price}$(ord1)) includes

(Person Price)

Person	Price
Fred	11.50
Fred	10.25
Fritz	12.25

Common part will be <Fred 10.25>

Union

r1 p∪ r2 = {t | t in r1 or t in r2}

ord1 p∪ ord2

ans(Person Price Amt)

	Person	Price	Amt
r1	Fred	unk	6
	Fred	10.25	unk
	Fritz	unk	unk
r2	Fred	11.50	6
	unk	10.25	7

What if We Let Intersection Be the Result?

Might only want tuples that are in every possible answer

pop(r) = {t|t in op(q) for every q in POSS(r)}

For example, r

PILOT	FLIGHT	DATE	TIME
Cushing	615	5Oct	**unk**
Cook	704	6Oct	5:50a
Cushing	**unk**	6Oct	**unk**

$p\pi_{PL}(r)$

PILOT	FLIGHT
Cushing	615
Cook	704

But Note

For-sure def. of operators does not guarantee For-sure semantics of queries with multiple operators.

r

PILOT	FLIGHT	DATE	TIME
Cushing	615	5Oct	**unk**
Cook	704	6Oct	5:50a
Cushing	**unk**	6Oct	**unk**

$p\sigma_{P=Cushing}(r) = \emptyset$

$p\pi_{LD}(p\sigma_{P=Cushing}(r)) =$

FLIGHT	DATE
615	5Oct

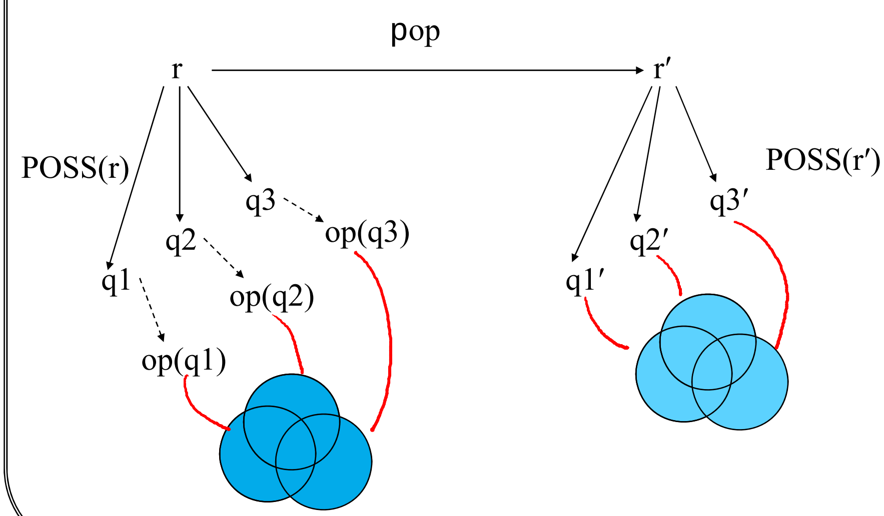
want this

"Could Be" Semantcs

Possibility Preserving: combination of all possible results

$\cup\{q \mid q \text{ in } POSS(pop(r))\} =$
$\cup\{op(q') \mid q' \text{ in } POSS(r)\}$

Diagram



Selection Conditions

t.B = 6

A	B	
<Fred	6>	true
<Fred	4>	false
<Fred	unk>	maybe

3-valued logic

or	T	F	M
T	T	T	T
F	T	F	M
M	T	M	M

Example

```
     A     B
t1  <Fred unk>
t2  <Fritz unk>

    t.A = Fritz or t.B = 6
t1        F     or   M     = M
t2        T     or   M     = T
```


But

```
t1  <Fred unk>

    t.B < 6 or t.B ≥ 6
t1        M     or   M     = M
```

But it would be T for any value for unk