Abstract

For now : I don't consider full-join. To do : A lot :) Don't forget to demonstrate that u dont loose certain answer that "normal" evaluation would return.

1 Preliminaries

Definition 1. *We denote the Set of well formed select query without agregation, full join and null test by* $[\![SQL]\!]$
We denote the Set of well formed select query without agregation and full join by $[\![SQL]\!]_\perp$

Definition 2. *Let's a Select query* $Q \in [\![SQL]\!]$ *a tuple* (Σ, R, H, P) *such that*

$$\begin{aligned}
&\Sigma \subseteq \{r_i.a_i | \exists r_i \in R \wedge \exists a_i \in r_i\}\\
&P \subseteq \{p_i = r_j.a_j | r_j \notin R\} \text{ a set of external parameter.}\\
&R \text{ a set of relation.}\\
&H \text{ belongs to the following grammar}
\end{aligned}$$

$$\begin{aligned}
H ::= {} & r_i.a_i = c_i \mid r_i.a_i \neq c_j \mid r_i.a_i = r_j.a_j \mid r_i.a_i \neq r_j.a_j \mid r_i.a_i = p_i \mid \\
& exists(Q) \mid notexists(Q) \mid in(r_i.a_i, Q) \mid notin(r_i.a_i, Q) \mid \\
& H \wedge H \mid H \vee H
\end{aligned}$$

Definition 3. *Let's a Select query* $Q \in [\![SQL]\!]_\perp$ *a tuple* (Σ, R, H, P) *such that*

$$\begin{aligned}
&\Sigma \subseteq \{r_i.a_i | \exists r_i \in R \wedge \exists a_i \in r_i\}\\
&P \subseteq \{p_i = r_j.a_j | r_j \notin R\} \text{ a set of external parameter.}\\
&R \text{ a set of relation.}\\
&H_\perp \text{ belongs to the following grammar}
\end{aligned}$$

$$\begin{aligned} H_\perp ::= &\ r_i.a_i = c_i \mid r_i.a_i \neq c_j \mid r_i.a_i = r_j.a_j \mid r_i.a_i \neq r_j.a_j \mid \\ &\ r_i.a_i = p_i \mid null(r_i.a_i) \mid const(r_i.a_i) \\ &\ exists(Q_\perp) \mid notexists(Q_\perp) \mid in(r_i.a_i, Q_\perp) \mid notin(r_i.a_i, Q_\perp) \mid \\ &\ H_\perp \wedge H_\perp \mid H_\perp \vee H_\perp \end{aligned}$$

We denote $(\Sigma, R, H, P)[x]$ the query $(\Sigma, R, H, P \cup x)$

Proposition 1.

$$[\![SQL]\!] \subset [\![SQL]\!]_\perp$$

Proof. immediate i guess ? □

Example 1. `SELECT F1.titre, F2.titre FROM Film as F1, Film as F2 WHERE NOT EXISTS`

```
(SELECT id_production FROM Production WHERE Production.id_film = F1.id_film
AND id_production NOT IN
(SELECT id_production FROM Production WHERE Production.id_film = F2.id_film))
and NOT EXISTS
(SELECT id_production FROM Production WHERE Production.id_film = F2.id_film
AND id_production NOT IN
(SELECT id_production FROM Production WHERE Production.id_film = F1.id_film));
```

–

$Q = (\Sigma, R, H)$
$\Sigma = \{F1.titre, F2.titre\}$
$R = \{F1, F2\}$
$H = notexists(Q_1) \wedge notexists(Q_2)$
$P = \emptyset$

–

$Q_1 = (\Sigma_1, R_1, H_1)$
$\Sigma_1 = \{id\_production\}$
$R_1 = \{Production\}$
$H_1 = Production.id\_film = F1.id\_film \wedge notin(id\_production, Q_{11})$
$P = \{F1.id\_film, F2.id_film\}$

–

$Q_{11} = (\Sigma_{11}, R_{11}, H_{11})$
$\Sigma_{11} = \{id\_production\}$
$R_{11} = \{Production\}$
$H_{11} = Production.id\_film = F2.id\_film$
$P = \{F1.id\_film, F2.id_film\}$

–

$Q_2 = (\Sigma_2, R_2, H_2)$
$\Sigma_2 = \{id\_production\}$
$R_2 = \{Production\}$
$H_2 = Production.id\_film = F2.id\_film \wedge notin(id\_production, Q_{21})$
$P = \{F1.id\_film, F2.id_film\}$

–
$Q_{21} = (\Sigma_{21}, R_{21}, H_{21})$
$\Sigma_{21} = \{id\_production\}$
$R_{21} = \{Production\}$
$H_{21} = Production.id\_film = F1.id\_film$
$P = \{F1.id\_film, F2.id_film\}$

1.1 Syntaxic Sugar

NO BAGS FOR NOW. (Only a sketch to keep it in mind)
AND WRONG

Proposition 2.

$$Eval_{SQL}((\Sigma_0, R_0 \cup (\Sigma_1, R_1, H_1, P_1), H_0, P_0), D) = Eval_{SQL}((\Sigma_0, R_0 \cup R_1, H_0 \wedge H_1, P_0 \cup P_1), D)$$

$$Eval_{SQL}((\Sigma_1, R_1, H_1, P_1), D) \cup Eval_{SQL}((\Sigma_2, R_2, H_2, P_2), D) = Eval_{SQL}((\Sigma_1, R_1 \cup R_2, H_1 \vee H_2, P_1 \cup P_2), D)$$

$$Eval_{SQL}((\Sigma_1, R_1, H_1, P_1), D) \cap Eval_{SQL}((\Sigma_2, R_2, H_2, P_2), D) = Eval_{SQL}((\Sigma_1, R_1 \cup R_2, H_1 \wedge H_2, P_1 \cup P_2), D)$$

$$Eval_{SQL}((\Sigma_1, R_1, H_1, P_1), D) \backslash Eval_{SQL}((\Sigma_2, R_2, H_2, P_2), D) = Eval_{SQL}((\Sigma_1, R_1, H_1 \wedge notexists((\Sigma_2, R_2, H_2 \wedge (\Sigma$$

$$Eval_{SQL}((\Sigma_0, R_0, notin(r_i.a_i, (\Sigma_1, R_1, H_1, P_1)), P_0), D) = Eval_{SQL}((\Sigma_0, R_0, notexists(\Sigma_1, R_1, H_1 \wedge r_i.a_i = \Sigma_1, P$$

$$Eval_{SQL}((\Sigma_0, R_0, in(r_i.a_i, (\Sigma_1, R_1, H_1, P_1)), P_0), D) = Eval_{SQL}((\Sigma_0, R_0, exists(\Sigma_1, R_1, H_1 \wedge r_i.a_i = \Sigma_1, P_1), P_0$$

Proof. To do .. □

2 Semantics

2.1 $Eval_{SQL}$

Definition 4.

$$\sigma_{\Sigma}(x) = (x[r_i.a_i] | r_i.a_i \in \Sigma)$$

Definition 5.

$$
\begin{aligned}
Eval_{SQL}((\Sigma, R, H_1 \wedge H_2, P), D) &= \llbracket \sigma_\Sigma(x)^n | x \in^n Eval_{SQL}((*, R, H_1, P), D) \wedge x \in^n Eval_{SQL}((*, R, H_2, P)\\
Eval_{SQL}((\Sigma, R, H_1 \vee H_2, P), D) &= \llbracket \sigma_\Sigma(x)^n | x \in^n Eval_{SQL}((*, R, H_1, P), D) \vee x \in^n Eval_{SQL}((*, R, H_2, P)\\
Eval_{SQL}((\Sigma, R, null(r_i.a_i), P), D) &= \llbracket \sigma_\Sigma(x)^n | x \in^n R, x[r_i.a_i] = \bot \rrbracket\\
Eval_{SQL}((\Sigma, R, const(r_i.a_i), P), D) &= \{\sigma_\Sigma(x) | x \in R, x[r_i.a_i] \neq \bot\}\\
Eval_{SQL}((\Sigma, R, r_i.a_i = c_i, P), D) &= \{\sigma_\Sigma(x) | x \in R, x[r_i.a_i] = c_i\}\\
Eval_{SQL}((\Sigma, R, r_i.a_i = r_j.a_j, P), D) &= \{\sigma_\Sigma(x) | x \in R, x[r_i.a_i] = x[r_j.a_j] \wedge x[r_i.a_i] \neq \bot \wedge x[r_j.a_j] \neq \bot\}\\
Eval_{SQL}((\Sigma, R, r_i.a_i = p_i, P), D) &= \{\sigma_\Sigma(x) | x \in R, x[r_i.a_i] = P[p_i] \wedge x[r_i.a_i] \neq \bot \wedge P[p_i] \neq \bot\}\\
Eval_{SQL}((\Sigma, R, r_i.a_i \neq c_i, P), D) &= \{\sigma_\Sigma(x) | x \in R, x[r_i.a_i] \neq c_i \wedge x[r_i.a_i] \neq \bot\}\\
Eval_{SQL}((\Sigma, R, r_i.a_i \neq r_j.a_j, P), D) &= \{\sigma_\Sigma(x) | x \in R, x[r_i.a_i] \neq x[r_j.a_j] \wedge x[r_i.a_i] \neq \bot \wedge x[r_j.a_j] \neq \bot\}\\
Eval_{SQL}((\Sigma, R, r_i.a_i \neq p_i, P), D) &= \{\sigma_\Sigma(x) | x \in R, x[r_i.a_i] \neq P[p_i] \wedge x[r_i.a_i] \neq \bot \wedge P[p_i] \neq \bot\}\\
Eval_{SQL}((\Sigma, R, exists(Q), P), D) &= \{\sigma_\Sigma(x) | x \in R, Eval_{SQL}(Q[x], D) \neq \emptyset\}\\
Eval_{SQL}((\Sigma, R, notexists(Q), P), D) &= \{\sigma_\Sigma(x) | x \in R, Eval_{SQL}(Q[x], D) = \emptyset\}
\end{aligned}
$$

Proposition 3.

$\forall Q \in \llbracket SQL \rrbracket, cert_\bot((\Sigma, R, H, P), D) = \{x | \forall h, h(x) \in Eval_{SQL}((\Sigma, R, H, h(P)), h(D)\}$

Proof. to do ! EvalSQL = EvalFO on complete database. and Q without null check. ☐

Proposition 4.

$\forall h, \forall Q \in \llbracket SQL \rrbracket (\{y | h(y) \in Eval_{SQL}(Q, h(D))\} = \emptyset \Rightarrow Eval_{SQL}(Q, h(D)) = \emptyset)$

Proof.

$$
\begin{aligned}
Eval_{SQL}(Q, h(D)) \neq \emptyset \Rightarrow\ & \exists x \in Eval_{SQL}(Q, h(D))\\
& \text{As } Eval_{SQL} \text{ has correctness guarentee on complete databases}\\
& \Rightarrow \exists x, x \in Cert_\bot(Q, h(D))\\
& \Rightarrow \exists x, \forall g, g(x) \in Eval_{SQL}(Q, g(h(D)))\\
& \Rightarrow \exists x, \text{especialy } g = h, h(x) \in Eval_{SQL}(Q, h(h(D)))\\
& \Rightarrow \exists x, h(x) \in Eval_{SQL}(Q, h(D))\\
& \Rightarrow \{y | h(y) \in Eval_{SQL}(Q, h(D))\} \neq \emptyset
\end{aligned}
$$

☐

3 Translation

$$
\begin{aligned}
(\Sigma, R, H, P)^+ &\rightarrow (\Sigma, R, H^*, P)\\
(\Sigma, R, H, P)^? &\rightarrow (\Sigma, R, H^{**}, P)
\end{aligned}
$$

$$
\begin{aligned}
(H_1 \wedge H_2)^* &\rightarrow H_1^* \wedge H_2^* \\
(H_1 \vee H_2)^* &\rightarrow H_1^* \vee H_2^* \\
(r_i.a_i = c_i)^* &\rightarrow r_i.a_i = c_i \\
(r_i.a_i \neq c_i)^* &\rightarrow r_i.a_i \neq c_i \\
(r_i.a_i = r_j.a_j)^* &\rightarrow r_i.a_i = r_j.a_j \\
(r_i.a_i \neq r_j.a_j)^* &\rightarrow r_i.a_i \neq r_j.a_j \\
null(r_i.a_i)^* &\rightarrow null(r_i.a_i) \\
const(r_i.a_i)^* &\rightarrow const(r_i.a_i) \\
exists(Q)^* &\rightarrow exists(Q^+) \\
notexists(Q)^* &\rightarrow notexists(Q^?) \\
in(r_i.a_i, Q)^* &\rightarrow in(r_i.a_i, Q^+) \\
notin(r_i.a_i, Q)^* &\rightarrow notin(r_i.a_i, Q^?)
\end{aligned}
$$

$$
\begin{aligned}
(H_1 \wedge H_2)^{**} &\rightarrow H_1^{**} \wedge H_2^{**} \\
(H_1 \vee H_2)^{**} &\rightarrow H_1^{**} \vee H_2^{**} \\
(r_i.a_i = c_i)^{**} &\rightarrow r_i.a_i = c_i \vee null(r_i.a_i) \\
(r_i.a_i \neq c_i)^{**} &\rightarrow r_i.a_i \neq c_i \vee null(r_i.a_i) \\
(r_i.a_i = r_j.a_j)^{**} &\rightarrow r_i.a_i = r_j.a_j \vee null(r_i.a_i) \vee null(r_j.a_j) \\
(r_i.a_i \neq r_j.a_j)^{**} &\rightarrow r_i.a_i \neq r_j.a_j \vee null(r_i.a_i) \vee null(r_j.a_j) \\
null(r_i.a_i)^{**} &\rightarrow null(r_i.a_i) \\
const(r_i.a_i)^{**} &\rightarrow const(r_i.a_i) \\
exists(Q)^{**} &\rightarrow exists(Q^?) \\
notexists(Q)^{**} &\rightarrow notexists(Q^+) \\
in(r_i.a_i, Q)^{**} &\rightarrow in(r_i.a_i, Q^?) \\
notin(r_i.a_i, Q)^{**} &\rightarrow notin(r_i.a_i, Q^+)
\end{aligned}
$$

Proposition 5.

$$\forall Q \in [\![SQL]\!], Eval_{SQL}(Q^+, D) \subseteq cert_\perp(Q, D)$$

Proposition 6.

$\forall Q \in [\![SQL]\!], posi_\perp(Q, D) = \{x | \exists h, h(x) \in Eval_{SQL}(Q, h(D))\} \subseteq Eval_{SQL}(Q^?, D)$

Proof. Assume (5).

By induction ...

$$
\begin{aligned}
x \in Eval_{SQL}((\Sigma, R, notexists(Q), P)^{+}, D) &\Rightarrow x \in Eval_{SQL}((\Sigma, R, notexists(Q^{?}), P), D) \\
&\Rightarrow Eval_{SQL}(Q^{?}[x], D) = \emptyset \\
&\Rightarrow \{y | \exists h, h(y) \in Eval_{SQL}(Q[h(x)], h(D))\} = \emptyset \text{ (by 5)} \\
&\Rightarrow \forall h, \forall y, h(y) \notin Eval_{SQL}(Q[h(x)], h(D)) \\
&\Rightarrow \forall h, \{y | h(y) \in Eval_{SQL}(Q[h(x)], h(D))\} = \emptyset \\
&\Rightarrow \forall h, Eval_{SQL}(Q[h(x)], h(D) = \emptyset \text{ (by 3)} \\
&\Rightarrow \forall h, h(x) \in Eval_{SQL}((\Sigma, R, notexists(Q), h(P)), h(D)) \\
&\Rightarrow x \in cert_{\perp}((\Sigma, R, notexists(Q), P), D)
\end{aligned}
$$

□

Proof. Assume (4).
By induction ...

$$
\begin{aligned}
x \in posi_{\perp}((\Sigma, R, notexists(Q), P), D) &\Rightarrow \exists h, h(x) \in Eval_{SQL}((\Sigma, R, notexists(Q), P), h(D)) \\
&\Rightarrow \exists h, Eval_{SQL}(Q[h(x)], h(D)) = \emptyset \\
&\Rightarrow \{y | \forall g, g(y) \in Eval_{SQL}(Q[g(x)], g(D))\} = \emptyset \\
&\Rightarrow Eval_{SQL}(Q^{+}[x], D) = \emptyset \text{ (by 4)} \\
&\Rightarrow x \in Eval_{SQL}(\Sigma, R, notexists(Q^{+}), P), D) \\
&\Rightarrow x \in Eval_{SQL}(\Sigma, R, notexists(Q), P)^{?}, D)
\end{aligned}
$$

□

4 Optimization

The translation $Q \rightarrow Q^{+}$ has an heavy cost has explained in the paper ..., in order to remove some useless null test we offer an optimization.

Definition 6. *For each Query and nested Query we maintain a set of attributes that have to be not null in order for the query to be true resp. false we denote this set* $\perp_Q^T$ *resp.*$\perp_Q^F$.

Definition 7.

$$
\begin{aligned}
\perp^T_{(\Sigma,R,H_1\wedge H_2,P)} &= \perp^T_{(\Sigma,R,H_1,P)} \cup \perp^T_{(\Sigma,R,H_2,P)}\\
\perp^T_{(\Sigma,R,H_1\vee H_2,P)} &= \perp^T_{(\Sigma,R,H_1,P)} \cap \perp^T_{(\Sigma,R,H_2,P)}\\
\perp^T_{(\Sigma,R,r_i.a_i=c_i,P)} &= \{r_i.a_i\}\\
\perp^T_{(\Sigma,R,r_i.a_i\neq c_i,P)} &= \{r_i.a_i\}\\
\perp^T_{(\Sigma,R,r_i.a_i=r_j.a_j,P)} &= \{r_i.a_i, r_j.a_j\}\\
\perp^T_{(\Sigma,R,r_i.a_i\neq r_j.a_j,P)} &= \{r_i.a_i, r_j.a_j\}\\
\perp^T_{(\Sigma,R,r_i.a_i=p_i,P)} &= \{r_i.a_i, p_i\}\\
\perp^T_{(\Sigma,R,r_i.a_i\neq p_i,P)} &= \{r_i.a_i, p_i\}\\
\perp^T_{(\Sigma,R,null(r_i.a_i),P)} &= \emptyset\\
\perp^T_{(\Sigma,R,const(r_i.a_i),P)} &= \{r_i.a_i\}\\
\perp^T_{(\Sigma,R,exists(Q),P)} &= \perp^T_{Q[?]}\\
\perp^T_{(\Sigma,R,notexists(Q),P)} &= \perp^F_{Q[?]}
\end{aligned}
$$

$$
\begin{aligned}
\perp^F_{(\Sigma,R,H_1\wedge H_2,P)} &= \perp^F_{(\Sigma,R,H_1,P)} \cap \perp^F_{(\Sigma,R,H_2,P)}\\
\perp^F_{(\Sigma,R,H_1\vee H_2,P)} &= \perp^F_{(\Sigma,R,H_1,P)} \cup \perp^F_{(\Sigma,R,H_2,P)}\\
\perp^F_{(\Sigma,R,r_i.a_i=c_i,P)} &= \emptyset\\
\perp^F_{(\Sigma,R,r_i.a_i\neq c_i,P)} &= \emptyset\\
\perp^F_{(\Sigma,R,r_i.a_i=r_j.a_j,P)} &= \emptyset\\
\perp^F_{(\Sigma,R,r_i.a_i\neq r_j.a_j,P)} &= \emptyset\\
\perp^F_{(\Sigma,R,r_i.a_i=p_i,P)} &= \emptyset\\
\perp^F_{(\Sigma,R,r_i.a_i\neq p_i,P)} &= \emptyset\\
\perp^F_{(\Sigma,R,null(r_i.a_i),P)} &= \emptyset\\
\perp^F_{(\Sigma,R,const(r_i.a_i),P)} &= \emptyset\\
\perp^F_{(\Sigma,R,exists(Q),P)} &= \perp^F_{Q[?]}\\
\perp^F_{(\Sigma,R,notexists(Q),P)} &= \perp^T_{Q[?]}
\end{aligned}
$$

(No projection again ... otherwise it's not as powerful as it should be : SELECT a FROM R WHERE a=b or null(B))

Proposition 7.

$$x \in Eval_{SQL}(Q,D) \Rightarrow \forall r_i.a_i \in \perp^T_Q, x^\sigma[r_i.a_i] \neq \perp$$

$$x \notin Eval_{SQL}(Q, D) \Rightarrow \forall r_i.a_i \in \perp_Q^F, x^\sigma[r_i.a_i] \neq \perp$$

Proof. □

If disjunction can do better.

Definition 8.

$$\begin{aligned}
nested^+(H_1 \wedge H_2) &= \{H_1 \wedge H_2\} \cup nested^+(H_1) \cup nested^+(H_2)\\
nested^+(H_1 \vee H_2) &= \{H_1 \vee H_2\} \cup nested^+(H_1) \cup nested^+(H_2)\\
nested^+(r_i.a_i = c_i) &= \{r_i.a_i = c_i\}\\
nested^+(r_i.a_i \neq c_i) &= \{r_i.a_i \neq c_i\}\\
nested^+(r_i.a_i = r_j.a_j) &= \{r_i.a_i = r_j.a_j\}\\
nested^+(null(r_i.a_i)) &= \{null(r_i.a_i)\}\\
nested^+(const(r_i.a_i)) &= \{const(r_i.a_i)\}\\
nested^+(exists(Q)) &= \{exists(Q)\} \cup nested^+(Q)\\
nested^+(notexists(Q)) &= \{notexists(Q)\} \cup nested^-(Q)
\end{aligned}$$

$$\begin{aligned}
nested^-(H_1 \wedge H_2) &= nested^-(H_1) \cup nested^-(H_2)\\
nested^-(H_1 \vee H_2) &= nested^-(H_1) \cup nested^-(H_2)\\
nested^-(r_i.a_i = c_i) &= \emptyset\\
nested^-(r_i.a_i \neq c_i) &= \emptyset\\
nested^-(r_i.a_i = r_j.a_j) &= \emptyset\\
nested^-(null(r_i.a_i)) &= \emptyset\\
nested^-(const(r_i.a_i)) &= \emptyset\\
nested^-(exists(Q)) &= nested^-(Q)\\
nested^-(notexists(Q)) &= nested^+(Q)
\end{aligned}$$

$$nested(Q) = nested^-(Q) \cup nested^+(Q)$$

Definition 9.

$$\begin{aligned}
notexists(Q')^{\perp}_{Q} &\rightarrow notexists(Q'^{\perp}_{Q})\\
exists(Q')^{\perp}_{Q} &\rightarrow exists(Q'^{\perp}_{Q})
\end{aligned}$$

Definition 10.

$$(\Sigma, R, H \vee null(r_i.a_i), P)^{\perp}_{Q} \to (\Sigma, R, H^{\perp}_{Q}, P)$$
$$\textit{if } \exists Q' \in nested^{+}(Q), (H \vee null(r_i.a_i)) \in nested(Q'), r_i.a_i \in \perp^{T}_{Q'}$$
$$\textit{if } \exists Q' \in nested^{-}(Q), (H \vee null(r_i.a_i)) \in nested(Q'), r_i.a_i \in \perp^{F}_{Q'}$$

Proposition 8.

$$Eval_{SQL}(Q, D) = Eval_{SQL}(Q^{\perp}_{Q}, D)$$

Proof.

Definition 11.

$$(\Sigma, R, H \vee null(r_i.a_i), P)^{\perp^F}_{Q} \to (\Sigma, R, H^{\perp^F}_{Q}, P)$$
$$\textit{if } \exists Q' \in nested^{-}(Q), (H \vee null(r_i.a_i)) \in nested(Q'), r_i.a_i \in \perp^{T}_{Q'}$$
$$\textit{if } \exists Q' \in nested^{+}(Q), (H \vee null(r_i.a_i)) \in nested(Q'), r_i.a_i \in \perp^{F}_{Q'}$$

Proposition 9.

$$\forall Q' \in nested^{+}(Q), Eval(Q, D) = Eval(Q^{\perp}_{Q'}, D)$$

Proposition 10.

$$\forall Q' \in nested^{-}(Q), Eval(Q, D) = Eval(Q^{\perp^F}_{Q'}, D)$$

Proof. Assume (10).
By Induction ...
$H_1 \wedge H_2 \in nested^{+}(Q)$
$\exists r_i.a_i \in (\perp^{T}_{H_1} \setminus \perp^{T}_{H_2}) \wedge (H \vee null(r_i.a_i) \in nested(H_2)$
Then if $x[r_i.a_i] = \perp$, $Eval(H^{T}_{2,H_2}, D, x)$ might be different from $Eval(H^{T}_{2,H_1 \wedge H_2}, D, x)$ □

□