4. Incremental Update Propagation
1. Principles of Update Propagation
   - Relational Algebra with Defining Equations
   - Fixpoint Semantics for Relational Algebra
   - Differential Calculus for Relations

2. Update Propagation in SQL
   - Basic SQL
   - Delta Views in SQL
   - Incremental Evaluation of Aggregates

3. Cost-based Update Propagation
   - Parameters of Cost Calculation
   - Magic Sets for Relational Algebra
   - Fixpoint Computation Using Soft Stratification
3 - Parameters of Cost Calculation

Example: • Suppose all relations P,Q,R are materialized

• Let |Q| = 30, |Q⁺| = 1, |R| = 20, |R⁺| = 2 , |P| = 200

• Let c be defined as follows: 
  \[ c(\{\text{delta views}\}) = \sum \text{op} \cdot w_{\text{op}} \cdot \min\{\text{para}\_\text{size}\} \]
  with \( w_{\pi} = 20, w_{\cup} = 30, w_{\setminus} = 10, \) and \( w_{\bowtie} = 40 \)

\[ P = \pi_X (Q \bowtie_C R) \]

medium focus Q⁺

\[ P⁺ = \pi_X (Q⁺ \bowtie_C R^{\text{new}}) \cup \pi_X (R^{\text{new}} \bowtie_C \text{Q}) \setminus P \]

c (medium focus Q⁺) = 40 * 1 + 40 * 22 + 20 * 1 + 20 * 22 + 10 * 22 + 30 * 1

\[ \equiv \ 1\text{-st join} \]
\[ \equiv \ 2\text{-nd join} \]
\[ \equiv \ 1\text{-st projection} \]
\[ \equiv \ 2\text{-nd projection} \]
\[ \equiv \ \text{difference} \]
\[ \equiv \ \text{union} \]

\[ = 1630 \]
3 - Magic Sets for Relational Algebra

\[ c(\{\text{delta views}\}) = \sum \text{op} \cdot w_{\text{op}} \cdot \min\{\text{para}\_\text{size}\} \]

- Having a nested algebra expression (e.g. \((R \cup S) \setminus P\)) and \(|R| << |S|,|P|\) the smallest relation size \(|R|\) dominates the costs of all operators (e.g. \(\cup\) and \(\setminus\)).

- This can only be justified if query optimization provides a strong focus on the smallest relation, e.g., by pushing selection conditions.

- **Problem**: In case of recursive views classical query optimization fails.

- Other forms of algebraic optimization techniques are required such as **Magic Sets**!
3 - Magic Sets for Relational Algebra

- Magic Sets is a transformation-based approach to query evaluation (Bancilhon et al. 86).
- Magic Sets represents a generalization of the *pushing selection* optimization.

original set of algebra equations:

\[ r_1: P = \sigma_{X=1}(Q \bowtie S) \]

\[ r_2: Q = \pi_{X,Y}(R) \]

corresponding RA operator tree:

\[ \sigma_{X=1} \]

\[ Q \bowtie S \]

\[ \pi_{X,Y} \]

\[ R \]

\[ S \]

selection condition
The idea is to introduce new relations (the magic sets) which contain selection constants and allow for gathering newly derived ones:

"original" pushing selection strategy

semi-joins instead of selections

pushed selection condition

containing selection condition Query_Q.X=1
Not only existing selections can be replaced by semi-joins but also newly derived ones so-called derived queries can be handled using this technique:

- semi-joins instead of selections
- semi-joins for derived selections

\[ \pi_{X,Y} \]
\[ \Join S \]
\[ \Join S \]
\[ \Join Q.Y=S.Y \]
\[ \Join Q.Y=S.Y \]
\[ \pi_{X,Y} \]
\[ \pi_{X,Y} \]
Evaluation of recursive relations can be enhanced by using dynamically generated selection conditions, too:

original set of algebra equations:

$$\begin{align*}
  r_1 & : A = \sigma_{X=1}(P) \\
  r_2 & : P = E \\
  r_3 & : P = E \bigcup \bigcap_{E.Y=P.X} P
\end{align*}$$

corresponding RA operator tree:
Evaluation of recursive relations can be enhanced by using dynamically generated selection conditions, too:

"original" RA operator tree:

semi-joins for derived selections

containing selection condition X=1
Going back to algebra equations yields the *Magic Sets transformed* rule set:

**semi-joins for derived selections**

\[
\begin{align*}
A &= \sigma_{X=1} (P) \\
P &= \text{Query}_P \bowtie E \\
P &= \text{Query}_P \bowtie E \bowtie P \\
\text{Query}_P &= \pi_Y (\text{Query}_P \bowtie E) \\
\text{Query}_P &= \text{Q}_\text{Seed}
\end{align*}
\]

**corresponding algebra equations:**

\[
\begin{align*}
r_1: &\quad A = \sigma_{X=1} (P) \\
r_2: &\quad P = \text{Query}_P \bowtie E \\
r_3: &\quad P = \text{Query}_P \bowtie E \bowtie P \\
r_4: &\quad \text{Query}_P = \pi_Y (\text{Query}_P \bowtie E) \\
r_5: &\quad \text{Query}_P = \text{Q}_\text{Seed}
\end{align*}
\]

more details are given in the current lecture on DDB by Pof. Manthey !!!
Graphical interpretation shows the enhanced evaluation for the query \( P(1,Y) ? \) when applying the transformed equations instead of the original ones:

**original set of algebra equations:**

**Magic Sets transformed equations:**

The selection constant 1 is not pushed into the recursive rule for \( P \) such that search is **not goal-directed!**

The selection constant 1 dynamically leads to further constants 2, 4, and 3 leading to a **goal-directed** search from node 1.

\[ \Rightarrow \text{paths } P(1,2), P(1,3), P(1,4) \text{ and } P(2,3) \text{ are generated only for } "P(1,Y)?" \]
3 - Magic Sets for Relational Algebra

- Magic Sets transformed equations allow for a goal directed evaluation of a given query.
- However, there are two major drawbacks of this transformation-based approach:
  - New relations and expensive join operations are introduced!
  - The resulting set of algebra equations may not be stratifiable anymore:

R:

\[
\begin{align*}
  r_1: & \quad P = B \setminus \pi_{X,Y}(Q) \setminus \pi_{Y,X}(Q) \\
  r_2: & \quad Q = D
\end{align*}
\]

The Magic Set transformed rule set \( R^{\text{magic}} \) for the query \( P(X,Y) \) is given by:

\[
\begin{align*}
  r_1: & \quad P = \text{Query}_P \bowtie B \setminus \pi_{X,Y}(Q) \setminus \pi_{Y,X}(Q) \\
  r_2: & \quad Q = \text{Query}_Q \bowtie D \\
  r_3: & \quad \text{Query}_Q = \pi_{Y,X}(\text{Query}_P \bowtie B \setminus \pi_{X,Y}(Q)) \ldots
\end{align*}
\]

unstratifiable
1. **Principles of Update Propagation**
   - Relational Algebra with Defining Equations
   - Fixpoint Semantics for Relational Algebra
   - Differential Calculus for Relations

2. **Update Propagation in SQL**
   - Basic SQL
   - Delta Views in SQL
   - Incremental Evaluation of Aggregates

3. **Cost-based Update Propagation**
   - Parameters of Cost Calculation
   - Magic Sets for Relational Algebra
   - **Fixpoint Computation Using Soft Stratification**
• This stratification problem is inherent, i.e., it cannot be avoided for certain rules.

• Unstratifiable equations introduce undefined facts in case of inconsistencies.

• However, all unstratifiable magic rules are good-natured, i.e., all facts are defined.

• For determining their semantics a new form of stratification is needed which takes the semantics of the compiled magic rules into account as well:

⇒ soft stratification (Behrend’03)
**3 - Fixpoint Computation Using Soft Stratification**

**stratification \( \lambda \):**
\[ \text{pred}(D) \rightarrow \mathbb{N} \]

1) \( p \) positively depends on \( q \) \( \Rightarrow \) \( \lambda(p) \geq \lambda(q) \)

2) \( p \) negatively depends on \( q \) \( \Rightarrow \) \( \lambda(p) > \lambda(q) \)

**iterated fixpoint computation wrt. \( \lambda \)**

(Apt/Blair/Walker,’86)

- **stratified rule set**
- **sequence of local fixpoint computations**

\[ \text{lfp}(T^*[R^#3], \text{lfp}(T^*[R^#2], \text{lfp}(T^*[R^#1], F))) \]

\[ \text{lfp}(t,f) = \text{least fixpoint of operator } t \text{ containing facts } f \]
weak stratification $\lambda$: $\text{pred}(D) \rightarrow \mathbb{N}$

1) $p$ positively depends on $q$ $\Rightarrow$ $\lambda(p) \geq \lambda(q)$
2) $p$ negatively depends on $q$ $\Rightarrow$ $\lambda(p) > \lambda(q)$

weakly stratified rule set

$\lambda(p)$:
1) $p$ positively depends on $q$ $\Rightarrow$ $\lambda(p) \geq \lambda(q)$
2) $p$ negatively depends on $q$ $\Rightarrow$ $\lambda(p) > \lambda(q)$

$lfp (T^{\text{weak}[R],F})$

new operator!
3 - Fixpoint Computation Using Soft Stratification

**Idea:** Integration of local fixpoint computations into the new operator $T^{\text{weak}}$ itself!

operator $T^{\text{weak}}[P]$ w.r.t. a given partition $P$ of $R$:

$$T^{\text{weak}}[P](f) := \begin{cases} f & \text{if } \neg \exists j : T^*[P \# j](f) \supseteq f \\ T^*[P \# j](f) & \text{otherwise} \end{cases}$$

**Note:** In a partition $P = P_1 \cup \ldots \cup P_n$ of $R$ every equation $r \in R$ is included in exactly one $P_i$ ($1 \leq i \leq n$).

A fixpoint computation using $T^{\text{weak}}$ and a stratified partition $P$ always leads to a correct determination of the semantics!!
Example of a fixpoint computation using $T^{\text{weak}}$ and a stratified rule set:
Observation: The course of computation corresponds exactly to one of the iterated fixpoint computation. This is always the case for stratified partitions!
'soft stratification' extends the definition of weak stratification by considering all negative and solely certain positive dependencies between magic rules $R^\text{magic}$:

$$H = \text{Query}_H \ op_1 L_1 \ op_2 \ldots \ op_n L_n \setminus P \setminus R$$

Each magic $R_1^m$ is complete wrt. its corresponding derived queries.

$$\Rightarrow \text{partial ordering of } R^m: \begin{pmatrix} \ L_1, R_1, R_2, R_2 \ & < \ & R_3^m \ R_1, R_1 \ & < \ & R_2^m \end{pmatrix} \ \text{soft stratification of } R^m$$
The application of soft stratification in our first example yields:

**R:**

- \( r_1: \ P = B \setminus \pi_{X,Y}(Q) \setminus \pi_{Y,X}(Q) \)
- \( r_2: \ Q = D \)

**R^{magic}:**

- \( r_1: \ P = \text{Query}_P \bowtie B \setminus \pi_{X,Y}(Q) \setminus \pi_{Y,X}(Q) \)
- \( r_2: \ Q = \text{Query}_Q \bowtie D \)
- \( r_3: \text{Query}_Q = \pi_{Y,X}(\text{Query}_P \bowtie B \setminus \pi_{X,Y}(Q)) \)
- \( r_4: \text{Query}_Q = \text{Query}_P \bowtie B \)

**Partition P of R^{magic} w.r.t. a soft stratification:**

- \( r_1: \ P = \text{Query}_P \bowtie B \setminus \pi_{X,Y}(Q) \setminus \pi_{Y,X}(Q) \) (P# 3)
- \( r_2: \ Q = \text{Query}_Q \bowtie D \) (P# 1)
- \( r_3: \text{Query}_Q = \pi_{Y,X}(\text{Query}_P \bowtie B \setminus \pi_{X,Y}(Q)) \) (P# 2)
- \( r_4: \text{Query}_Q = \text{Query}_P \bowtie B \)

\( \text{lfp} (\text{T}^{\text{weak}}[P], F) = \) correct answers for P