on query rewriting

what logical propeties are usefull for static analysis?

outline

1. logical properties of the algebraic operators (see e.g., notes of CS245 from Stanford U.)
2. rules for rewriting subqueries
3. containment of conjunctive queries

subqueries

example:
StarsIn[movieTitle,movieYear,starName]
MovieStar[name, address, gender, birthdate]

```
SELECT movieTitle
FROM StarsIn
WHERE starName IN (
SELECT name FROM MovieStar WHERE birthdate = 1960)
```

two-argument selection

$$\sigma(R,\varphi) = \{t \in R | \varphi(t) = true\}$$

with

- R is a relation
- φ is a complex condition on R

example

$$\pi_{movieTitle}(\sigma(StarsIn,(StarName\ IN\ \pi_{name}(\sigma_{birthdate\ =\ 1960}(MovieStar)))))$$

rewriting a two-argument selection

the rewriting depends on

- the condition (IN, NOT IN, EXISTS, etc.)
- the correlation between the outer query and the subquery

uncorrelated IN conditions

rewriting rule:

$$\sigma(R,(t\ IN\ S))$$
$$\equiv$$
$$\sigma_C(R \times \delta(S))$$

where:

- t stands for a (possibly projected) tuple of R
- C is the condition that equates t to the tuples in S

example

$$\pi_{movieTitle}(\sigma(StarsIn,(StarName\ IN\ \pi_{name}(\sigma_{birthdate\ =\ 1960}(MovieStar)))))$$

rewrites

$$\pi_{movieTitle}(StarsIn \bowtie_{starName=name} \pi_{name}(\sigma_{birthdate\ =\ 1960}(MovieStar)))$$

(δ omitted since *name* is the key for *MovieStar*)

handling correlated subqueries

problem: subquery involves unknown values defined outside themselves

principle:

- add extra attributes to the subquery
- relate extra attribute to the inner attributes with selection condition
- do not forget to project out extra attribute when no longuer necessary
- do not forget to eliminate duplicates when necessary

example

find the movies where the average age of the stars was at-most 40 when the movie was made

```
SELECT DISTINCT m1.movieTitle, m1.movieYear
FROM StarsIn m1
WHERE m1.movieYear - 40 <= (
    SELECT AVG(birthdate)
    FROM StarsIn m2, MovieStar s
    WHERE m2.starName = s.name
    AND m1.movieTitle = m2.movieTitle
    AND m1.movieYear = m2.movieYear)
```

algebraic formulation with a two-argument selection

$$\delta(\pi_{m1.movieTitle,m1.movieYear}(\sigma(StarsIn\ m1,(m1.movieYear - 40 <= \gamma_{AVG(s.birthdate)}(\sigma_{m2.movieTitle=m1.movieTitle \wedge m2.movieYear=m1.movieYear}(StarsIn\ m2 \bowtie_{m2.starName=s.name} MovieStar\ s))))))$$

- $\sigma_{m2.movieTitle=m1.movieTitle \wedge m2.movieYear=m1.movieYear}$ must be deferred until after the combination with StarsIn m1
- attributes $m2.movieTitle, m2.movieYear$ must be available after the γ

without two-argument selection

$$\delta(\pi_{m1.movieTitle,m1.movieYear}(\sigma_{m1.movieYear-40<=avg}(\\ StarsIn\ m1 \bowtie_{m2.movieTitle=m1.movieTitle \wedge m2.movieYear=m1.movieYear} \\ \gamma_{m2.movieTitle,m2.movieYear,AVG(s.birthdate)\rightarrow avg}((\\ StarsIn\ m2 \bowtie_{m2.starName=s.name} MovieStar\ s))))))$$

in addition, note that:

- starNames from m1 are projected out
- the join involving m1 gives the same title and year as in m2

after applying other rewriting rules

$$\delta($$
$$|$$
$$\pi_{m2.movieTitle,m2.movieYear}($$
$$|$$
$$\sigma_{m2.movieYear-40<=avg}($$
$$|$$
$$\gamma_{m2.movieTitle,m2.movieYear,AVG(s.birthdate)\rightarrow avg}($$
$$|$$
$$StarsIn\ m2 \bowtie_{m2.starName=s.name}\ MovieStar\ s))))))$$

containement of conjunctive queries

example: let Q_1 and Q_2 be two conjunctive queries

SELECT	R1.B, R1.A	SELECT	R3.A, R1.A
FROM	R R1, R R2	FROM	R R1, R R2, R R3
WHERE	R2.A=R1.B	WHERE	R1.B=R2.B AND R2.B=R3.A

put differently

$Q_1 = \pi_{2,1}(\sigma_{2=3}(R \times R))$

$Q_2 = \pi_{5,1}(\sigma_{2=4 \wedge 4=5}(R \times R \times R))$

or even

$Q_1(x,y) \leftarrow R(y,x),R(x,z)$

$Q_2(x,y) \leftarrow R(y,x),R(w,x),R(x,u)$

examples

are Q_1 and Q_2 equivalent?

if yes, processing Q_1 saves one join

can classical algebraic rewriting rules be used?

no!

query equivalence and query containment

definitions : given 2 queries q and q' on a schema D

- $q \subset q'$ if for all instance I of D, $q(I) \subset q'(I)$
- $q \equiv q'$ if $q \subset q'$ and $q' \subset q$

substitution

recall that a *valuation* is

- a function from $var(q)$ to dom
- extended to free tuples

now, for a conjunctive query q, a *substitution* is

- a function from $var(q)$ to $var \cup dom$
- extended to free tuples

example

consider Q_2 and substitution θ such that

- $\theta(x) = x$
- $\theta(y) = y$
- $\theta(u) = z$
- $\theta(w) = y$

applying θ to Q_2 yields:
$Q_2(x,y) \leftarrow R(y,x),R(y,x),R(x,z)$ that is Q_1

query containment

there exists a substitution that transforms the body of Q_2 into the body of Q_1

if I is an instance and $t \in Q_1(I)$

there exists a valuation v applied to Q_1 that leads to t

therefore $v \circ \theta$ is a valuation that applied to Q_2 leads to t

therefore $t \in Q_2(I)$ which shows that $Q_1(I) \subset Q_2(I)$ and thus Q_1 is contained in Q_2

example

let $I(R) = \{(1,2),(2,3)\}$

consider the valuation $v(y) = 1, v(x) = 2, v(z) = 3$

thus $t = (2,1) \in Q_1(I)$

consider now the valuation $\theta' = v \circ \theta$

we have
$\theta'(w) = \theta'(y) = v(y) = 1, \theta'(x) = v(x) = 2, \theta'(u) = v(z) = 3$

we have $t = (2,1) \in Q_2(I)$

homomorphism

let q and q' be two rules on the same database schema B

an *homomorphism* from q' to q is:

- a substitution θ such that
- $\theta(body(q')) \subseteq body(q)$ and $\theta(tete(q')) = tete(q)$

the homomorphism theorem

let q and q' be two queries on the same schema

$q \subseteq q'$ if there exists an homomorphism from q' to q

corollary: two queries q and q' on the same schema are equivalent if

- there exists an homomorphism from q to q' and
- there exists an homomorphism from q' to q

complexity

the test of query equivalence is

- a problem in *NPTIME* for conjunctive queries
- an *undecidable* problem for relational queries

practically

How is rewriting taken into account in your favorite RDBMS?