Query Optimization: Exercise

Session 1

Bernhard Radke

October 22, 2018

- Exercise sessions are here to illustrate the material of the course with examples, special cases, etc.
- Homework every week: programming assignment and a few problems
- Do 75% or better to get a bonus of 0.3 on your exam grade
- Written exam at the end of the semester (presumably on Feb. 26)
- Slides on the website (db.in.tum.de/teaching/ws1819/queryopt)

Algebra Revised

uni schema:

- Studenten : {[MatrNr: integer, Name: string, Semester: integer]}
- Vorlesungen : {[VorlNr: integer, Titel: string, SWS: integer, gelesenVon: integer]}
- Professoren : {[PersNr: integer, Name: string, Rang: string, Raum: integer]}
- Assistenten : {[PersNr: integer, Name: string, Fachgebiet: string, Boss: integer]}
- hoeren : {[MatrNr: integer, VorlNr: integer]}
- voraussetzen : {[Vorgaenger: integer, Nachfolger: integer]}
- pruefen : {[MatrNr: integer, VorlNr: integer, PersNr: integer, Note: decimal]}

Relational Calculus

- *what* the result looks like (declarative)
- tuple calculus: $\{t|P(t)\}$
  - $\{p|p \in \text{Professoren} \wedge p.Rang = \text{'C4'}\}$
  - $\{s|s \in \text{Studenten}$

    $\wedge \exists h \in \text{hoeren}(s.MatrNr = h.MatrNr$

    $\wedge \exists v \in \text{Vorlesungen}(h.VorlNr = v.VorlNr$

    $\wedge \exists p \in \text{Professoren}(p.PersNr = v.gelesenVon \wedge p.Name = \text{'Curie'}))))\}$
- domain calculus: $\{[v_1, ..., v_n]|P(v_1, ..., v_n)\}$
  - $\{[p, n, r, o]|[p, n, r, o] \in \text{Professoren} \wedge r = \text{'C4'}\}$
  - $\{[m, n, s]|\exists m([m, n, s] \in \text{Studenten}$

    $\wedge \exists v([m, v] \in \text{hoeren}$

    $\wedge \exists p([v, t, d, p] \in \text{Vorlesungen}$

    $\wedge \exists a([p, a, r, o] \in \text{Professoren} \wedge a = \text{'Curie'})))\}$

- compare that to SQL
  - `SELECT * FROM Professoren p WHERE p.Rang='C4'`
  - ```
    SELECT s.MatrNr, s.Name, s.Semester
    FROM Studenten s, hoeren h, Vorlesungen v, Professoren p
    WHERE s.MatrNr=h.MatrNr AND h.VorlNr=v.VorlNr AND
          v.gelesenVon=p.PersNr AND p.Name='Curie'
    ```
- *what* the result looks like (declarative)
    ```

Relational Algebra

- *how* the result is built (procedural)
  - $\sigma_{Rang='C4'}(\text{Professoren})$
  - $\sigma_{S.MatrNr=H.MatrNr}(S \times \sigma_{H.VorlNr=V.VorlNr}($
    $H \times \sigma_{V.gelesenVon=P.PersNr}($
    $V \times \sigma_{P.Name='Curie'}(P))))$
  - $S \bowtie (H \bowtie (V \bowtie_{V.gelesenVon=P.PersNr} \sigma_{P.Name='Curie'}(P)))$

Textbook Optimization

- Translate SQL into an executable plan
- Many equivalent plans
- Large differences in resource consumption
- Minimize cost function
- $$C_{\text{out}}(T) = \begin{cases} 0 & \text{if } T \text{ is a leaf } R_i \\ |T| + C_{\text{out}}(T_1) + C_{\text{out}}(T_2) & \text{if } T = T_1 \bowtie T_2 \end{cases}$$

- Find all Students that attend the course 'Ethik'
  - SQL query
  - canonical translation
  - break up conjunctive selections
  - push down selections
  - introduce joins
  - determine join order
  - introduce and push down projections

Programming Assignments

TinyDB

- very simple database system
- storage layer and runtime system already implemented
- you will build a compile time system step by step
- Code: fork gitlab.db.in.tum.de/qo18/tasks

Homework Guidelines

- General
  - You can work in groups of up to two students
  - Include a file containing your names, email addresses and matriculation numbers into your fork.
  - Handwritten (and/or scanned) solutions will not be accepted! Use LaTeX (preferable) or Word.
  - Push as PDF into your gitlab fork
- Programming
  - Target: GNU/Linux
  - Language: c++ (great opportunity to learn it)
  - Submissions:
    - You can work within the TinyDB directory, changing its structure if needed
    - (Briefly) comment the code: every class, field, method, design choice

- Slides: db.in.tum.de/teaching/ws1819/queryopt
- Exercise task: gitlab
- Questions, Comments, etc:
  mattermost @ mattermost.db.in.tum.de/qo18
- Exercise due: 9 AM next monday