

3. Anfrageverarbeitung

- ❑ Übersetzungsverfahren für Datenbanksprachen
 - Vorgehensweise
 - Übersetzung vs. Interpretation
- ❑ Anfrageoptimierung
 - Anfragedarstellung
 - Anfragetransformation
 - Erstellung und Auswahl von Zugriffsplänen

 16.

Vorgehensweise bei der Anfrageverarbeitung

1. Parsen der Anfrage (lexikalische und syntaktische Analyse)
 - Überführen der Anfrage in eine Interndarstellung
2. Semantische Analyse
 - Ersetzen der Sichten durch ihre relationalen Ausdrücke.
 - Gibt es die in der Anfrage angesprochenen Relationen und Attribute tatsächlich in der Datenbank?
 - Zugriffskontrolle
3. Normalisierung der Anfrage
 - Überführung in eine Normalform
 - Vereinfachung der Anfrage, d. h. Erkennen von Redundanzen und leeren Teilanfragen
4. Erstellung von Ausführungsplänen und Auswahl des besten Ausführungsplans

 18.

Problem:

- ❑ Gegeben eine SQL-Anfrage. Erzeuge aus der Anfrage einen günstigen (ideal wäre einen optimalen) physischen Operatorbaum.

Schwierigkeiten bei der Transformation

- ❑ Eigenschaften von SQL
 - nicht-prozedurale (deskriptive) Anfragesprache
 - zugriffspfad-unabhängiges (relationales) Datenmodell
 - Fakten und Beziehungen werden durch Werte dargestellt
 - Zugriff auf Satzmenge
- ❑ Physischer Operatorbaum
 - prozedurale Darstellung einer Anfrage
 - Ausnutzung von Indexstrukturen
 - Zugriff auf einzelne Datensätze

 17.

3.1 Parsen einer Anfrage

- ❑ Aufgabe des Parsers:
Transformiere eine SQL Anfrage in einen Baum dessen Knoten sich auf folgende Objekte beziehen:
 - **Terminalsymbole:** lexikalische Elemente von SQL wie z. B. Schlüsselwörter (SELECT), Operatoren, Klammern, Namen von Attributen und Relationen.
 - **Variablen:** Namen für in einer Anfrage syntaktische ähnlichen Teil einer Anfrage. Diese Variablen werden durch eine Zeichenkette repräsentiert, die mit einem Paar von dreieckigen Klammern umgeben ist.

Beispiele:

<SFW> steht für die Anfragen mit den typischen Klauseln select, from und where.

<Condition> repräsentiert die Bedingungen in der where-Klausel.

 19.

Grammatik

- ❑ Wir stellen im folgenden kurz eine Grammatik für einen sehr stark eingeschränkten Sprachumfang von SQL vor.
 - Schlüsselwörter group by, having, order by, distinct, union, natural join werden nicht unterstützt.
 - ...

Anfragen

- ❑ $\langle \text{Query} \rangle ::= \langle \text{SFW} \rangle$
- ❑ $\langle \text{Query} \rangle ::= (\langle \text{Query} \rangle)$

Select-From-Where Ausdruck

- ❑ $\langle \text{SFW} \rangle ::= \text{SELECT } \langle \text{SelList} \rangle \text{ FROM } \langle \text{FromList} \rangle \text{ WHERE } \langle \text{Condition} \rangle$

Select Liste

- ❑ $\langle \text{SelList} \rangle ::= \langle \text{Attribute} \rangle, \langle \text{SelList} \rangle$
- ❑ $\langle \text{SelList} \rangle ::= \langle \text{Attribute} \rangle$

20.

Beispiel:

- ❑ Wir betrachten die folgenden beiden Relationen:
 - InLand(LName, SName)
 - Stadt(Name, Einw)
- ❑ Wir sind nun an den Ländernamen interessiert, in der eine Stadt mit mehr als 500,000 Einwohner existiert.
 - `SELECT LName
FROM InLand
WHERE Sname IN (SELECT Name FROM Stadt WHERE Einw > 500000)`
 - `SELECT LName
FROM InLand, Stadt
WHERE Einw > 500000 AND Name = SName`

22.

From Liste

- ❑ $\langle \text{FromList} \rangle ::= \langle \text{Relation} \rangle, \langle \text{FromList} \rangle$
- ❑ $\langle \text{FromList} \rangle ::= \langle \text{Relation} \rangle$

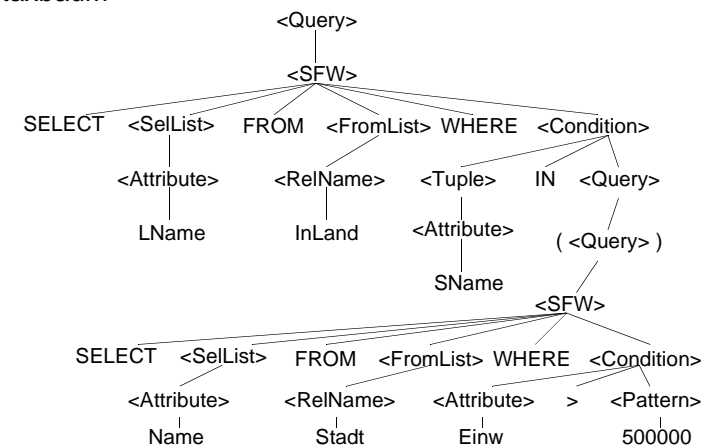
Bedingungen

- ❑ $\langle \text{Condition} \rangle ::= \langle \text{Condition} \rangle \text{ AND } \langle \text{Condition} \rangle$
- ❑ $\langle \text{Condition} \rangle ::= \langle \text{Tuple} \rangle \text{ IN } \langle \text{Query} \rangle$
- ❑ $\langle \text{Condition} \rangle ::= \langle \text{Attribute} \rangle = \langle \text{Attribute} \rangle$
- ❑ $\langle \text{Condition} \rangle ::= \langle \text{Attribute} \rangle > \langle \text{PATTERN} \rangle$

Variablen wie z. B. $\langle \text{Attribute} \rangle$, $\langle \text{Relation} \rangle$ und $\langle \text{Pattern} \rangle$ repräsentieren eine Menge von Atomen und sind selbst nicht durch eine Regel definiert.

21.

Syntaxbaum



23.

Semantische Analyse

- ❑ Ersetzen aller Sichten in der from-Klausel durch ihre Anfragen
 - Expansion eines Knotens im Baum
- ❑ Überprüfen, ob die Relationen tatsächlich existieren.
 - Gibt es die Relationen InLand und Stadt in der Datenbank?
- ❑ Überprüfen, ob die Attribute in der Datenbank existieren und Ergänzung des zugehörigen Relationennamen vor dem eigentlichen Attributnamen. Gibt es keine eindeutige Zuordnung, wird eine Fehlermeldung ausgegeben.
 - Das Attribut LName wird erkannt und durch InLand.LName ersetzt.
- ❑ Typüberprüfung bei den Attributen
 - Sind InLand.SName und Stadt.Name vom gleichen Typ?

24.

Logische Operatoren

- ❑ Neben den Operatoren der relationalen Algebra, ist es sinnvoll noch weitere Operatoren der erweiterten relationalen Algebra (siehe DBS I) zu betrachten:
 - $|X|$: Verbundoperationen wie z. B. der natürliche Verbund
 - δ : Duplikateliminierung
Dieser Operator ist von zentraler Bedeutung, da im Gegensatz zur relationalen Algebra Anfragen in relationalen DBMS i. A. Multimengen verarbeitet werden.
 - π : (verallgemeinerte) Projektion (Map)
Für jedes Tupel der Eingaberelation wird genau ein Tupel erzeugt. Jedes Attribut der Ausgaberektion wird durch einen Ausdruck berechnet, der von den Attributen der Eingaberelation abhängt.

Beispiel:

Sei $R(A,B,C)$ eine Relation. Dann wird durch

$\pi_{A*B \rightarrow D, C \rightarrow E}$

eine Relation mit zwei Attributen D und E erzeugt. Sind Attribute der Eingaberelation identisch mit der Ausgaberektion, können wir die übliche Notation der Projektion benutzen. Z. B.

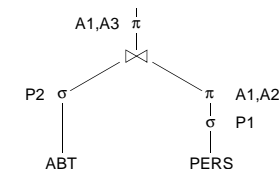
$\pi_{A*B \rightarrow D, C}$

26.

3.2 Transformation des Syntaxbaums

Interndarstellung einer Anfrage

- ❑ In der Datenbankliteratur finden sich viele Vorschläge für Anfragedarstellungen (siehe z. B. das Buch von B. Mitschang)
- ❑ Eine auf einem Operatorgraphen basierende Darstellung erscheint für die spätere Transformation der Anfrage am geeignetsten.
 - Knoten stellen *logische* Operatoren der relationalen Algebra dar
 - Kanten beschreiben operator-kontrollierten Datenfluß
- ❑ Beispiel:



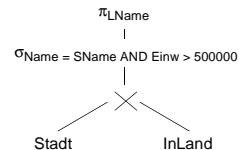
25.

- γ : Gruppierung und Aggregation
Der Operator arbeitet auf einer Relation R. Es werden zu einer Menge von Attributen von R die Äquivalenzklassen (bzgl. Gleichheit einer ausgewählten Menge von Attributen) gebildet und auf die Tupel der Klassen eine oder mehrere Aggregatsfunktionen angewendet. Die genaue Notation ist dann $\gamma_{A1,...,An,agg1 \rightarrow B1,...,aggm \rightarrow Bm}(R)$
Wird keine Aggregatsoperation angegeben und sind alle Attribute von R als Parameter aufgeführt, so führt der Operator eine Duplikateliminierung durch.

27.

Regeln für die Transformation

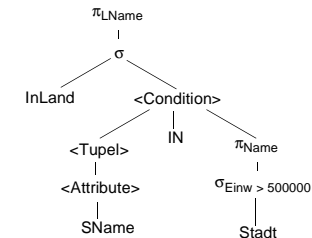
- ❑ Falls eine Anfrage eine einfache SWF-Anfrage ohne Unteranfrage ist, kann diese direkt in relationale Algebra umgesetzt werden:
 - Erzeuge das kartesische Produkt der Relationen in der from-Klausel
 - Schränke das Ergebnis auf die Tupel ein, welche die Bedingung der where-Klausel erfüllen.
 - Projektion des Ergebnis auf die in der select-Klausel angegebenen Attribute



28.

Transformation von Unteranfragen

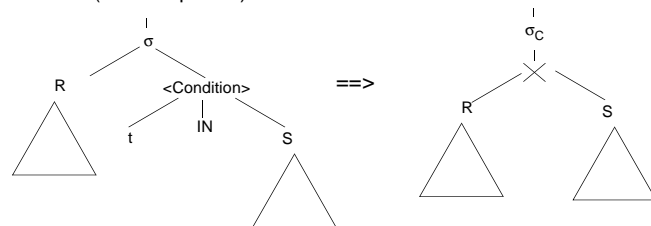
- ❑ Definition eines zweistelligen Selektionsoperators ohne Parameter. Der Operator repräsentiert im linken Kind die zugrundeliegende Relation und als rechtes Kind die Bedingung (in Form der Unteranfrage).
- ❑ Beispiel:



- ❑ Ziel ist es nun, den zweistelligen Selektionsoperator zu ersetzen.

29.

- ❑ Ist die Unterfrage vom Typ "t IN S" und S ist nicht korreliert (wie dies bei unserem Beispiel der Fall ist), so kann die folgende Ersetzungsstrategie durchgeführt werden:
 - Ersetze den Syntaxbaum von S durch einen Ausdruck in relationaler Algebra. Falls S Duplikate besitzt, muß noch eine Duplikateliminierung ausgeführt werden (siehe erweiterte relationale Algebra in DBS I).
 - Ersetze den zweistelligen Selektionsoperator durch einen einstelligen mit Bedingung C, wobei C die Attribute des Tupels und die entsprechenden von C auf Gleichheit testet.
 - Als Argument von dem einstelligen Operator dient das kartesische Produkt von R und S (ohne Duplikate).



- ❑ Bei korrelierten Unteranfragen vom Typ "t IN S" ist dies etwas komplexer (siehe z. B. das Buch Garcia-Mollina).

30.

Beispiel für korrelierte Unteranfragen

- ❑ Relationen: Stadt(SName, Einw), InLand(LName, SName)
 Anfrage:
 Gib die Namen der Städte deren Einwohnerzahl 100,000 über dem Durchschnitt der Einwohnerzahl eines Landes liegt.

```

select SName
from Stadt S, InLand L
where L.SName = S.SName and
      Einw - 100000 > select avg(Einw)
                      from Stadt, InLand
                      where InLand.SName = Stadt.SName and
                            L.LName = InLand.LName

```

Diese korrelierte Unteranfrage kann nun in folgende Anfrage transformiert werden:

31.

```

select SName
from Stadt, InLand,
      select LName as X, avg(Einw) as LEinw
      from Stadt, InLand
      where InLand.SName = Stadt.SName
      group by LName
where Einw - 100000 > LEinw and
      Stadt.SName = InLand.SName and
      InLand.LName = X

```

Die prinzipielle Vorgehensweise ist korrelierte Bedingung aus der Unteranfrage in die darüberliegende Anfrage als Joinbedingung hochzuziehen.

32.

Vereinfachung von Anfragen

Ziel bei der Anfrageübersetzung ist das frühzeitige Erkennen von

- ☐ leeren Anfragen
- ☐ gemeinsamen Teilanfragen

Vereinfachung der where-Klausel

- ☐ Beseitigung redundanter Prädikate
Gefahr redundanter Prädikate durch Integritätsregeln, Sichten, Zugriffsrechten, ...

Beispiel einiger Regeln der Boole'schen Algebra:

| | | |
|----------------|-----|------|
| A OR A | ⇔ | A |
| A AND (A OR B) | ⇔ | A |
| A OR FALSE | ⇔ | A |
| A OR TRUE | ⇔ | TRUE |
| ... | ... | ... |

34.

3.3 Algebraische Anfrageoptimierung

Normalisierung der Prädikate in der where-Klausel

- ☐ Durch Anwendung der Kommutativ-, Assoziativ- und Distributivregeln sowie der De Morgan'schen Regeln läßt sich die where-Klausel in folgende Formen übertragen:
 - konjunktive Normalform
($P_{i1} \text{ OR } P_{i2} \text{ OR } \dots P_{in}$) AND ... AND ($P_{j1} \text{ OR } P_{j2} \text{ OR } \dots P_{jm}$)
 - disjunktive Normalform
($P_{k1} \text{ AND } P_{k2} \text{ AND } \dots P_{kp}$) OR ... OR ($P_{l1} \text{ AND } P_{l2} \text{ AND } \dots P_{lp}$)
 Die Prädikate P_g sind dabei atomar.
- ☐ Bemerkungen
 - OR-Verknüpfung stellt eine Vereinigung von Mengen dar
 - AND-Verknüpfungen entspricht dabei der Schnittbildung (d. h. Join)

33.

Behandlung von Formeln mit leeren Mengen

| | | |
|---|---|------------|
| FORALL r IN {} AND P(r) | ⇔ | TRUE |
| EXISTS r IN {} and P(r) | ⇔ | FALSE |
| $M = \{r \mid r \text{ IN } \{\} \text{ and } P(r)\}$ | ⇔ | $M = \{\}$ |

Hüllenbildung

- ☐ Zu einer gegebenen Ausdruck werden nun weitere Prädikate redundant erzeugt.
- ☐ Beispiel:
 - where p.Ort = "Marburg" AND a.Ort = p.Ort AND j.Ort = p.Ort
 - Diese where-Klausel kann nun um zwei weitere atomare Formeln ergänzt werden:
where ... AND a.Ort = "Marburg AND j.Ort = "Marburg".
- ☐ Es ist offensichtlich, daß durch dieses Propagieren der Konstanten die Kosten der Anfragebearbeitung verringert werden.

35.

Berücksichtigung von Integritätsregeln

- Integritätsregeln sind Invarianten (Prädikate) der Datenbank der Form $A \Rightarrow B$.
- Beispiel
 - Gegeben eine Relation P (Personal) mit Attributen Ort, Gehalt und Alter.
 - Es gelten folgende Integritätsregeln:
 - a) $P.\text{Alter} < 20 \Rightarrow P.\text{Gehalt} < 2000$
 - b) $P.\text{Alter} > 40 \Rightarrow P.\text{Gehalt} > 4000$

Anfrage:

```
select  Name
from    Personal
where   Ort = "MR" and Alter > 30 and Gehalt < 4000
```

Durch Hinzufügen der zweiten Integritätsregeln läßt sich die Anfrage vereinfachen:

```
where   Ort = "MR" and Alter > 30 and Gehalt < 4000 and
        (Alter ≤ 40 or Gehalt > 4000)
```

Damit läßt sich die where-Klausel zu

```
where   Ort = "MR" and Alter > 30 and Alter ≤ 40 and Gehalt < 4000
umformen.
```

36.

Selektion

- Selektionen sollen möglichst frühzeitig ausgeführt werden, da
 - der Aufwand linear in der Anzahl der Tupel ist und
 - die Anzahl der Ergebnisse niedriger ist als die Anzahl der Eingabedaten.
- Folgende Regeln können angewendet werden:
 - $\sigma_F(R \cup S) = \sigma_F(R) \cup \sigma_F(S)$
 - $\sigma_F(R - S) = \sigma_F(R) - \sigma_F(S) = \sigma_F(R) - S$
 - $\sigma_F(R \times S) = \sigma_F(R) \times S$, falls die Attribute aus F auch in dem Schema von R liegen.
 - $\sigma_{F_1 \wedge F_2}(R) = \sigma_{F_1}(\sigma_{F_2}(R)) = \sigma_{F_2}(\sigma_{F_1}(R))$
 - $\sigma_{F_1 \vee F_2}(R) = \sigma_{F_1}(R) \cup \sigma_{F_2}(R)$

38.

Join

- Die Vereinigung, das kartesische Produkt und der natürliche Verbund sind
 - kommutativ ($R \bowtie X \bowtie S = S \bowtie X \bowtie R$) und
 - assoziativ ($(R \bowtie X \bowtie S) \bowtie X \bowtie T = R \bowtie X \bowtie (S \bowtie X \bowtie T)$).

Aufgrund dieser Eigenschaften können sehr viele äquivalente Umformungen von Operatorbäumen berechnet werden.

- Die Eigenschaft der Assoziativität gilt aber i. A. nicht für den Theta-Join.

Seien R, S und T Relationen mit A,C aus RS(R), B aus RS(S) und D aus RS(T).

Dann ist:

$(R \bowtie_{A>B} S) \bowtie_{C>D} T$ definiert, aber $R \bowtie_{A>B} (S \bowtie_{C>D} T)$ nicht (da das Attribut keinen Bezug zur Relation S und T besitzt).

37.

Projektion

- Die (allg.) Projektion reduziert nicht die Anzahl der Attribute (da keine Duplikateliminierung vorgenommen wird), sondern verändert (i. A. reduziert) die Anzahl der Attribute.
- Eine Projektion kann jederzeit zusätzlich in einem Operatorbaum verwendet werden, falls die herausprojizierten Attribute in den Nachfolgeknoten nicht mehr benötigt werden.

Einige Regeln:

Wir betrachten zwei Relationen R und S mit $X \subseteq RS_R$ und $Y \subseteq RS_S$.

- $\pi_Z(R \bowtie X \bowtie S) = \pi_Z(\pi_X(R) \bowtie \pi_Y(S))$, wobei $X \cup Y \supseteq Z$ und $X \cap Y = \emptyset$.

Man darf also auf die Argumente eines Joins zusätzlich eine Projektion ausführen, wenn die Joinoperation nicht dadurch verändert wird und alle benötigten Attribute für den darauffolgenden Operator vorhanden sind.

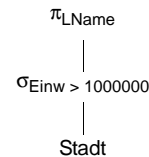
- $\pi_Z(\sigma_F(R)) = \pi_Z(\sigma_F(\pi_X(R)))$, wobei $X \supseteq Z$ und X alle Attribute enthält, die in der Formel F angesprochen werden.

39.

Das Herunterdrücken einer Projektion (insbesondere unter eine Selektion) kann aber auch zu einem Plan mit höheren Kosten führen.

Beispiel:

Wir betrachten die Relation Stadt(Name,LName,Einw) und fragen nach allen Ländern, die eine Stadt mit mehr als 1000000 Einwohner enthalten.



Liegt z. B. ein Index auf dem Attribut Einw der Relation Stadt, so könnte die Selektion durch den Index sehr schnell beantwortet werden. Durch Einführung einer weiteren Projektion $\pi_{\text{LName, Einw}}(\text{Stadt})$ würde dies nicht mehr möglich sein.

Die Frage, ob eine Projektion sinnvoll ist, hängt aber auch von der Anzahl der Antworten bei der Selektion ab. Qualifiziert sich fast jedes Tupel (womit der Vorteil des Index verloren geht), so könnte es von Vorteil sein, eine Projektion wiederum frühzeitig auszuführen.

40.

Ein Vertauschen der Duplikateliminierung mit den Operatoren Vereinigung, Minus und Projektion ist nicht möglich, wobei wir hierbei annehmen, dass Vereinigung und Minus auf Multimengen definiert sind.

Beispiel:

- ❑ Die Relation $\delta(R \cup S)$ kann für jedes Tupel nur eine Instanz besitzen, während die Relation $\delta(R) \cup \delta(S)$ auch zwei Instanzen haben kann.
- ❑ Betrachten wir die Relation T(A,B) mit der Instanz $\{(1,2), (1,3)\}$. Dann ist
 - $\delta(\pi_A(T)) = \{1\}$ und
 - $\pi_A(\delta(R)) = \{1, 1\}$.

42.

Duplikateliminierung

- ❑ Durch Duplikateliminierung wird die Anzahl der Tupel reduziert, was eine möglichst frühe Anwendung nahe legt.
- ❑ Im Gegensatz zur Selektion ist die Beseitigung von Duplikaten wesentlich aufwendiger und erfordert superlinearen Aufwand (siehe späteres Kapitel).
- ❑ Eine Duplikateliminierung löst sich auf $(\delta(R) = R)$, wenn z. B.
 - ein Attribut von R Primärschlüssel oder vom Typ unique ist,
 - oder direkt vorher eine Duplikateliminierung durchgeführt wurde.

Einige Regeln:

- ❑ $\delta(R \times S) = \delta(R) \times \delta(S)$
- ❑ $\delta(R \mid X \mid S) = \delta(R) \mid X \mid \delta(S)$
- ❑ $\delta(\sigma_F(R)) = \sigma_F(\delta(R))$

Aufgrund der höheren Kosten für die Duplikateliminierung ist die Frage berechtigt, ob ein Vertauschen hier tatsächlich zu niedrigeren Kosten führt.

41.

Gruppieren und Aggregation

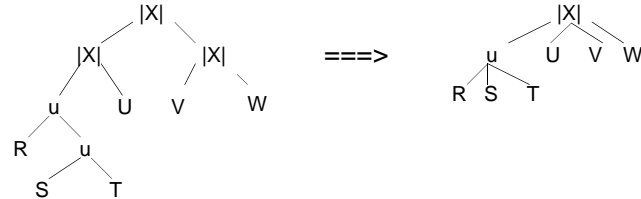
Folgende Regeln können verwendet werden:

- ❑ $\delta(\gamma_X(R)) = \gamma_X(R)$
Gruppieren beseitigt bereits die Duplikate!
- ❑ $\gamma_X(R) = \gamma_X(\pi_Y(R))$, wobei Y alle Eingabeattribute von X beinhaltet.
- ❑ Die Umformung $\gamma_X(R) = \gamma_X(\delta(R))$ kann nur bei Aggregaten wie Minimum und Maximum angewendet werden.

43.

Zusammenfassung von Operatoren

- Die assoziativen und kommutativen Operatoren vom gleichen Typ können innerhalb eines Operatorbaums zu mehrstelligen Operatoren zusammengefaßt werden.



- Die Berechnung einer geeigneten Reihenfolge wird auf eine spätere Phase bei der Optimierung verschoben.

44.

Kostenbasierte Anfragestransformation

Transformation des zuvor hergeleiteten logischen Operatorbaums in einen physischen Operatorbaum auf Basis eines einfachen und effektiven Kostenmodells.

1. Festlegen einer effizienten Reihenfolge der mehrstelligen Operatoren (hier insbesondere Verbund).
2. Auswahl einer effizienten Implementierung für die Operatoren:
3. Zusammenfügen von aufeinanderfolgenden Operatoren zu neuen Operatoren. So ist es z. B. möglich, eine Selektion und eine Projektion (ohne Duplikateliminierung) in einem Operator durchzuführen.
4. Festlegung eines Kommunikationsprotokoll zwischen zwei aufeinanderfolgenden Knoten im Operatorbaum. Wir werden hier annehmen, dass alle Operatoren die ONC-Schnittstelle erfüllen und die Ergebnisse der Teilbäume bedarfsorientiert berechnet werden. Die Alternative dazu wäre alle Ergebnisse eines Teilbaums zu berechnen und auf Platte zu speichern, die dann wieder vom Nachfolger eingelesen werden.

46.

Zusammenfassung

Folgende Schritte werden bei der logischen Anfrageoptimierung stets durchgeführt:

1. Zerlege eine Selektionen mit zusammengesetzten Prädikatstermen in mehrere Selektionen mit jeweils einem Term.
2. Führe Selektionen so früh wie möglich aus.
3. Fasse aufeinanderfolgende Selektionen derselben Relation zusammen. Diese Regel ist invers zu der Regel 1.
4. Führe Projektionen so früh wie möglich aus (ohne Duplikate zu eliminieren).
5. Fasse aufeinanderfolgende Projektionen derselben Relation zusammen.

Zusätzlich können noch die oben diskutierten Strategien betrachtet werden.

45.

Kostenmodell

Vorgehensweise:

- Es werden mehrere physische Operatorbäume erzeugt und auf Basis eines Kostenmodells miteinander verglichen.
- Dabei wird zunächst das Problem der Berechnung einer geeigneten Reihenfolge bei den mehrstelligen Joinoperatoren betrachtet.

Metadaten

Für eine Basisrelation R sollen folgende Daten zusätzlich verwaltet werden:

- $T(R)$: Anzahl der Tupel
- $V(R, A) = T(\delta(\pi_A(R)))$

Zusätzlich von Interesse sind auch:

- $B(R)$: Anzahl der Blöcke auf dem Externspeicher, in denen Tupel von R liegen.
- $V(R, \{A_1, \dots, A_n\}) = T(\delta(\pi_{A_1, \dots, A_n}(R)))$.

47.

Annahmen:

Die Kosten eines Operatorbaums werden durch die Anzahl der Zwischenergebnisse der Operatoren bestimmt.

Vorgensweise:

- ☐ Um eine geeignete Reihenfolge für die Joinberechnung zu finden, werden die oben genannten Metadaten für die Eingaberelationen benötigt. Diese liegen aber i. A. nicht vor, da die Eingaberelationen nicht physisch existieren, sondern zur Laufzeit berechnet werden.
- ☐ Somit müssen wir uns nun überlegen, wie wird diese Werte geeignet schätzen können. Wir betrachten dabei die einzelnen Operatoren der rel. Algebra und schätzen unter der Annahme, dass die Werte für die Eingaberelationen vorliegen, die entsprechenden Werte der Ausgabe.

48.

Selektion

Fallunterscheidungen für $\sigma_F(R)$

- ☐ F ist eine einfache atomare Formel vom Typ: "A = c", "A != c" oder "A > c"
- ☐ F ist eine komplexe Formel, die mit **and** und **or** verknüpft ist.

1. Fall: "A = c"

Es wird hierbei die Annahme getroffen, dass der Wert c tatsächlich in der Relation R vorkommt. Dann gilt:

$$T = T(\sigma_{A=c}(R)) = T(R) / V(R,A)$$

$$V = V(\sigma_{A=c}(R), A) = 1$$

Für die Berechnung von $V(\sigma_{A=c}(R), B)$ mit $B \neq A$ gilt:

$V(R,B)$ ist die Anzahl der Werte des Attributs B in der Relation R. Die Wahrscheinlichkeit, dass ein Tupel aus $\sigma_{A=c}(R)$ einen der Werte annimmt ist unter der Annahme der Gleichverteilung $1/V(R,A)$. Somit ist also

$$1 - 1/V(R,A)$$

die Wahrscheinlichkeit, dass dieser Wert nicht angenommen wird.

50.

Projektion

Aufgrund der verallg. Definition der Projektion gilt offensichtlich:

- ☐ $T(\pi_X(R)) = T(R)$
- ☐ $V(\pi_X(R), A) = V(R,A)$
für alle Attribute aus R, die auch Attribute der Ausgaberektion sind.

Wird ein Attribut B der Ausgaberektion über einen Ausdruck definiert, so ist es i. A. sehr schwierig, eine genaue Schätzung von $V(\pi_X(R), B)$ anzugeben. Es gibt aber einige Spezialfälle:

- ☐ $c \rightarrow B$: Attribut B ist konstant, d. h. $V(\pi_X(R), B) = 1$
- ☐ $A * c + d \rightarrow B$ ($c \neq 0$): Attribut B hängt linear von einem Attribut A der Relation R ab. Dann ist $V(\pi_X(R), B) = V(R,A)$.

49.

Dieses "Experiment" wird nun T-mal wiederholt. Somit ist also die Wahrscheinlichkeit, dass ein Wert des Attributs B nicht angenommen wird

$$(1 - 1/V(R,A))^T$$

und somit die Wahrscheinlichkeit, dass es mindestens einmal angenommen wird:

$$1 - (1 - 1/V(R,A))^T$$

Die Anzahl der Werte kann dann durch

$$V(\sigma_{A=c}(R), B) = T * (1 - (1 - 1/V(R,A))^T)$$

abgeschätzt werden (Formel von Cardenas).

2. Fall: "A > c"

$$T(\sigma_{A > c}(R)) = T(R) / 2$$

Dieser Abschätzung basiert auf der Annahme der Gleichverteilung. In System R wurde aufgrund der Erfahrungswerte als Teiler 3 verwendet!

Weiterhin gilt:

$$V(\sigma_{A > c}(R), A) = V(R,A)/2$$

Für alle anderen Attribute solle die Formel von Cardenas benutzt werden.

51.

3. Fall: "A != c"

$$T(\sigma_{A \neq c}(R)) = T(R) * (V(R,A) - 1) / V(R,A)$$

$$V(\sigma_{A \neq c}(R), A) = V(R,A) - 1$$

4. Fall: F = "F1 and F2"

Da $\sigma_{F1 \text{ and } F2}(R) = \sigma_{F1}(\sigma_{F2}(R))$, können wir diesen Fall auf zwei einfachere Fälle zurückführen. Man sollte hierbei schon vorher die Formeln so vereinfacht haben, dass z. B. leere Ausdrücke beseitigt wurden (z. B. der Form $\sigma_{A = 10 \text{ and } A > 20}(R)$).

Beispiel:

Sei eine Relation $R(A,B,C)$ mit $T(R) = 10000$ und $V(R,A) = 50$ gegeben. Wir betrachten die Anfrage $Q = \sigma_{A=10 \text{ and } B < 20}(R)$. Dann gilt

$$T(Q) = T(R) / (50 * 2) = 100$$

$$V(Q,A) = 1$$

52.

Join

- ☐ Der Fall des allgemeinen Joins mit einer Bedingung der Form "A > B" wird auf die Selektion zurückgeführt:

$$T(\sigma_{A > B}(R \times S)) = T(R) * T(S) / 2$$
- ☐ Wir betrachten im folgenden ausführlich den Fall für den natürlichen Verbund. Dabei nehmen zunächst einmal an, dass die Joinbedingung auf einem Attribut A definiert ist. Somit gibt es Relationen $R(X,A)$ und $S(A,Y)$ mit $X \cap Y = \emptyset$.
- ☐ Die Anzahl der Tupel eines Joins kann extrem schwanken:
 - Sind R und S bzgl. des Attributs A disjunkt. Dann gilt $T(R \mid X \mid S) = 0$.
 - Ist A ein Schlüssel von S und ein Fremdschlüssel von R. Dann gilt:

$$T(R \mid X \mid S) = T(R)$$
 - Wenn alle Tupel den gleichen Wert bzgl. A besitzen, gilt:

$$T(R \mid X \mid S) = T(R) * T(S)$$

54.

5. Fall: F = "F1 or F2"

Vereinfachende Annahme: F1 und F2 sind unabhängig voneinander.

Sei $Q1 = \sigma_{F1}(R)$ und $Q2 = \sigma_{F2}(R)$. Dann ist

$$T(\sigma_{F1 \text{ or } F2}(R)) = T(R) * (1 - (1 - T(S1)/T(R)) * (1 - T(S2)/T(R)))$$

Die Anzahl der Werte kann z. B durch folgende Formel geschätzt werden:

$$V(\sigma_{F1 \text{ or } F2}(R), A) = \min(V(Q1,A) + V(Q2,A), V(R,A))$$

Diese Formel überschätzt i. A. den konkreten Wert, da nicht berücksichtigt wird, dass gleiche Werte bei Q1 und Q2 für Attribut A angenommen werden.

53.

Vereinfachende Annahmen

- ☐ Falls zwei Relationen R und S ein Attribut A gemeinsam besitzen, gilt eine der folgenden Bedingungen:
 - $\pi_A(R) \supseteq \pi_A(S)$
 - $\pi_A(S) \supseteq \pi_A(R)$
- ☐ Sei B kein Joinattribut, das zur Relation R, aber nicht zur Relation S gehört. Dann gilt:

$$V(R \mid X \mid S, B) = V(R,B)$$

Bemerkungen:

- ☐ Die erste Annahme ist erfüllt, wenn A ein Schlüssel in R und ein Fremdschlüssel in S ist, oder umgekehrt. Diese Annahme ist in vielen Fällen nicht erfüllt!
- ☐ Entsprechend ist auch die zweite Annahme dann erfüllt, wenn das Joinattribut ein Schlüssel in S und ein Fremdschlüssel in R ist. Die Annahme ist dann nicht erfüllt, wenn es "dangling tuples" in R gibt.

55.

Sei $V(R,A) \leq V(S,A)$. Unter den oben gemachten Voraussetzungen wird dann jedes Tupel aus R die Joinbedingung mit $T(S) / V(S,A)$ Tupeln aus S eingehen (siehe auch Fall 1 bei der Selektion). Somit ist die Anzahl der Tupel bei einem Join durch

$$T(R) \cdot T(S) / V(S,A)$$

gegeben. Gilt nun $V(R,A) \geq V(S,A)$, kann entsprechend argumentiert werden. Somit gilt:

$$T(R \mid X \mid S) = T(R) \cdot T(S) / \max(V(S,A), V(R,A))$$

Joins mit mehreren Attributen

- Wir betrachten jetzt den Fall, wenn $\{A_1, \dots, A_n\}$ im Schema von R und S liegen.
- Neben den oben gemachten Annahmen wird weiter angenommen, dass die Attribute unabhängig voneinander sind.
- Die Wahrscheinlichkeit, dass zwei Tupel die Joinbedingung bzgl. dem Attribut A_i eingehen, ist dann

$$sel_i = 1 / \max(V(R, A_i), V(S, A_i))$$
- Die Wahrscheinlichkeit, dass alle Bedingungen erfüllt sind, ist gerade $sel_1 \cdot \dots \cdot sel_n$. Somit ist also

$$T(R \mid X \mid S) = T(R) \cdot T(S) \cdot sel_1 \cdot sel_2 \cdot \dots \cdot sel_n$$

56.

Andere Operatoren

- Die Schätzung der Kenngrößen ist äußerst schwierig für andere Operatoren!
- Duplikatbeseitigung $Q = \delta(R)$
 - Die Größe des Resultats kann zwischen 1 und $T(R)$ schwanken. Deshalb wird empfohlen, $T(Q) = T(R)/2$ zu setzen.
 - Weiterhin gilt: $T(Q) \leq V(R, A_1) \cdot V(R, A_2) \cdot \dots \cdot V(R, A_n)$.
- Gruppieren und Aggregation
 - Es wird hier auch wieder ein ähnliches Vorgehen wie bei der Duplikatbeseitigung empfohlen.

58.

Joins mit mehreren Relationen

- Wir betrachten jetzt den Join $Q = R_1 \mid X \mid R_2 \mid X \mid \dots \mid X \mid R_n$, wobei ein Attribut A in k der Relationen vorkommt (o. B. d. A.: R_1, R_2, \dots, R_k).
- Weiterhin sei $v_i = V(R_i, A)$ und es gelte $v_1 \leq v_2 \leq \dots \leq v_k$. Darüberhinaus nehmen wir wieder an, dass $\pi_A(R_1) \subseteq \pi_A(R_2) \subseteq \dots \subseteq \pi_A(R_k)$ gilt. Wir betrachten ein Tupel t aus R_1 . Dann ist die Wahrscheinlichkeit für t einen passenden Partner in R_i zu finden gerade $1/v_i$.

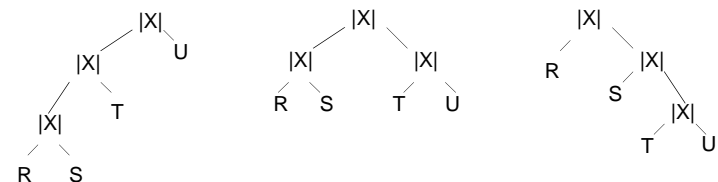
Satz

Die Schätzung für die Größe eines Joins ist unabhängig von der Reihenfolge.

57.

Suchstrategien

- Sei n die Anzahl der aufeinanderfolgenden Joinoperationen. Dann ergibt sich sofort, daß die Größe des Suchraums sehr schnell anwächst.
 - Dabei gibt es alleine $n!$ Möglichkeiten die Blattknoten zu variieren.
 - Multiplikativ geht dann noch die Anzahl der verschiedenen strukturierten Bäume ein.



- Es ist deshalb nicht machbar, alle möglichen Pläne zu betrachten, sondern approximativ nach Lösungen zu suchen.

59.

- Um das Optimierungsproblem zu lösen, beschränkt man sich häufig auf einen Teilraum der Lösungen der Form

$$\text{Join}(\text{Join}(\dots\text{Join}(\text{Join}(R_{i_1}, R_{i_2}), R_{i_3}), \dots), R_{i_n})$$

wobei (i_1, \dots, i_n) eine Permutation des Vektors $(1, \dots, n)$ ist. Der Grund hierfür liegt im wesentlichen an den Algorithmen für die Verarbeitung von Joins.

- Lässt man auch Lösungen zu, bei denen die innere Relation ebenfalls das Ergebnis eines Joins sein kann, so spricht man von **buschartigen Verbunden** (engl: bushy joins).
- Strategien zur Berechnung einer Reihenfolge
 - Enumerative Verfahren
 - Dynamische Programmierung
 - Heuristische Strategien (Greedy-Verfahren, zufallsgesteuerte Algorithmen, simuliertes Ausglühen, ...)

60.

Beispiel:

- Relationen $R(A,B)$, $S(B,C)$, $T(C,D)$ und $U(D,A)$ mit folgenden Kenngrößen:
 - $T(R) = T(S) = T(T) = T(U) = 1000$
 - $V(R,A) = 100$, $V(R,B) = 200$
 - $V(S,B) = 100$, $V(S,C) = 500$
 - $V(T,C) = 20$, $V(T,D) = 50$
 - $V(U,A) = 50$, $V(U,D) = 1000$

- Ergebnisse im 2. Schritt:

| | {R,S} | {R,T} | {R,U} | {S,T} | {S,U} | {T,U} |
|-------------|---------|-----------|---------|---------|-----------|---------|
| Größe | 5.000 | 1.000.000 | 10.000 | 2.000 | 1.000.000 | 1.000 |
| Kosten | 0 | 0 | 0 | 0 | 0 | 0 |
| Bester Plan | R X S | R X T | R X U | S X T | S X U | T X U |

- Ergebnisse im 3. Schritt:

| | {R,S,T} | {R,S,U} | {R,T,U} | {S,T,U} |
|-------------|-----------------|-----------------|-----------------|-----------------|
| Größe | 10.000 | 50.000 | 10.000 | 2.000 |
| Kosten | 2.000 | 5.000 | 1.000 | 1.000 |
| Bester Plan | (S X T) X R | (R X S) X U | (T X U) X R | (T X U) X S |

62.

Dynamisches Programmieren

- Dieser Ansatz wurde bereits in System R verwendet.

Idee:

- Berechne eine Reihenfolge für den Join $\{R_1, \dots, R_n\}$ durch Berechnung des besten Plans für

$$\text{Join}(\{R_1, R_2, \dots, R_{i-1}, R_{i+1}, \dots, R_n\}) \mid X \mid R_i, i = 1, \dots, n$$
- Die Kosten eines Plans werden durch die Summe der Zwischenergebnisse berechnet.
- Bottom-up Vorgehensweise
 - Wir starten zunächst mit den einzelnen Relationen R_i : Kosten 0 und Anzahl der Ergebnisse $T(R_i)$.
 - Dann betrachten wir das Problem $\text{Join}(\{R_i, R_j\})$, ordnen jedem Paar Kosten 0 zu (keine Zwischenergebnisse) und berechnen noch die Anzahl der Ergebnisse.
 - Danach berechnen wir die besten Pläne für $n = 3, 4, \dots$

61.

Methoden zur Selektivitätsschätzung

- Unsere bisherigen Ansätze beruhen auf einer Schätzung der Anzahl der Zwischenergebnisse durch Verwendung von Kenngrößen unter der Annahme der Gleichverteilung.
- Fragen:
 - Können wir bessere Schätzmethoden bekommen, indem wir weitere Metadaten abspeichern?
 - Was sind zu vorgegebenem Speicherplatz die besten Metadaten, um den Schätzfehler zu minimieren?
 - Wie teuer ist die Erzeugung der Metadaten? Was sind zu vorgegebenen Erzeugungskosten die besten Metadaten, um den Schätzfehler zu minimieren?

63.

Stichproben (Sampling)

Motivation

- Ähnlich zu der Vorhersage eines Wahlergebnisses kann man direkt oder indirekt mit Stichproben versuchen, die Kosten einer Anfrage zu schätzen.

Definition

- Unter einer gleichverteilten Stichprobe einer Relation R verstehen wir eine Teilmenge von m Datensätzen, $m < |R|$, so daß ein Datensatz aus R mit Wahrscheinlichkeit $m/|R|$ in der Stichprobe liegt.

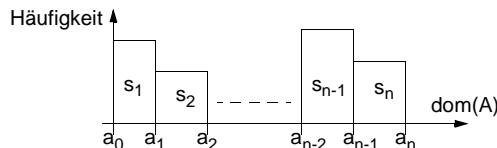
Probleme

- Elemente der Stichprobe sollen bereits dann gezogen werden, wenn die Eingabe noch nicht vollständig vorliegt.
- Ziehen einer Stichprobe kostet $O(m)$ Seitenzugriffe. Alternativ kann ein blockbasiertes Stichprobenverfahren benutzt werden.
- Stichprobengröße muß sehr hoch sein, um eine genaue Schätzung zu bekommen.

64.

Histogramme

- Sei R eine Relation, A ein Attribut der Relation und n eine ganze Zahl, $n > 0$. Eine disjunkte und vollständige Unterteilung des Datenraums $D = \text{dom}(A)$ in n Intervalle $I_1 = [a_0, a_1)$, ..., $I_n = [a_{n-1}, a_n)$ wird als Histogramm bezeichnet, falls zu jedem Intervall I_j eine Schätzung $s(I_j)$ für die Anzahl der Tupel vorliegt.
- Vorgehen zum Aufbau eines Histogramms
 - Ziehe eine Stichprobe aus der Datenbank.
 - Berechne die Anzahl der Tupel in I_j , $j = 1, \dots, n$.



66.

Blockbasiertes Stichprobenverfahren

Idee

- Idealerweise wird durch ein blockbasiertes Stichprobenverfahren $O(m/B)$ Zugriffe benötigt (m = Stichprobengröße, B = Anzahl der Datensätze pro Block)

Problem

- Datensätze innerhalb eines Blocks können unabhängig, teilweise korreliert oder sogar vollständig korreliert sein.

Methode der Crossvalidierung

- Nimm eine Stichprobe der Größe m durch Einlesen von m/B Blöcken;
- FOR ($i=1$, ähnlich = false; !ähnlich; $i++$)
 - a) Nimm eine Stichprobe der Größe $2^{i-1} \cdot m$ durch Einlesen von $2^{i-1} \cdot m/B$ Blöcke;
 - b) ähnlich = gezogene Stichprobe ist zu der vorhandenen Stichprobe ähnlich;
 - c) Verschmelze die beiden Stichproben miteinander;

65.

Abschätzung der Kardinalität

- Für ein Intervall $[a, b)$ definieren wir $|[a, b)| = b - a$.
- Bereichsanfrage mit Bereich $[l, r]$

$$\sum_{j=1}^n \frac{|[l, r] \cap [a_{j-1}, a_j)|}{|[a_{j-1}, a_j)|} \times s_j$$

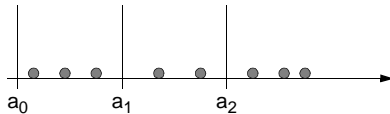
Bemerkungen

- Innerhalb eines Intervalls des Histogramms wird die Annahme getroffen, daß die Daten gleichverteilt sind.
- Durch Erhöhung der Anzahl der Intervalle ergibt sich i. a. keine Verbesserung der Schätzung. Die optimale Anzahl der Intervalle wird insbesondere durch die Stichprobengröße bestimmt.
- Durch Histogramme ergibt sich i. a. eine genauere Schätzung bzgl. des absoluten Fehlers bzw. des relativen Fehlers.

67.

Varianten von Histogrammen

- ☐ **equi-width**-Histogramm
Alle Intervalle I_j haben die gleiche Länge h .
- ☐ **equi-depth**-Histogramm
Alle Intervalle enthalten die gleiche Anzahl von Datensätzen.
- ☐ **max-diff**-Histogramm
Berechne die $n-1$ Paare von benachbarten Punkten mit maximalen Abstand. Ziehe zwischen solch einem Paar eine Intervallgrenze ein.



Mehrdimensionale Histogramme

- ☐ Abschätzung von Anfragen mit Prädikaten, die sich auf mehrere Attribute einer Relation beziehen.
- ☐ Statt in Intervalle wird der Datenraum in disjunkte Rechtecke zerlegt.
- ☐ Bei einer Bereichsanfrage wird der Flächeninhalt des Schnitts zwischen Suchrechteck und den Rechtecken des Histogramms berechnet.

68.

Optimierungszeitpunkte

Zentrale Frage

- ☐ Wie steht die Anfrageübersetzung (inkl. Optimierung) zur Anfrageausführung?

vollständige Vorübersetzung:

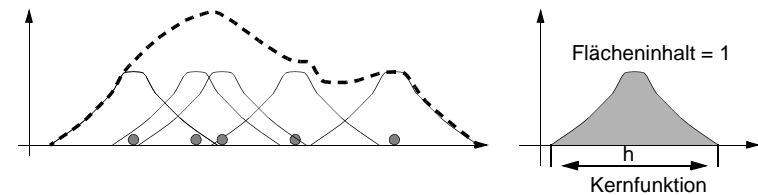
- ☐ strikte Trennung der Übersetzung einer Anfrage und ihrer Ausführung
 - schnellere Ausführung durch
 - a) aufwendigere Optimierung
 - b) wenig Zugriffe auf Metadaten
 - langsamere Ausführung durch
 - a) Änderungen der Datenbank (z. B. neuer Index, Größe der Datenbank)
- ☐ Optimierungskosten amortisieren sich bei mehrfacher Ausführung der Anfrage
 - z. B. in einer Schleife eines AWP
- ☐ Invalidierung des Zugriffsmoduls (Löschen eines Index) führt zur Neuübersetzung vieler Anfragen.

70.

Kernschätzer

- ☐ ist eine in der Statistik verwendete Methode, um die Dichte einer Verteilungsfunktion abzuschätzen.
- ☐ Die Konvergenz (in Abhängigkeit der Stichprobengröße) ist schneller als die bei Verwendung von Histogrammen.
- ☐ Um Schätzwerte für die Kardinalität zu berechnen, muß die Dichtefunktion im Suchintervall integriert werden.

Idee

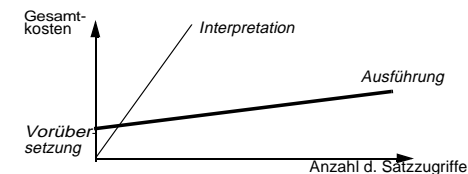


- ☐ Jeder Stichprobenpunkt ist Zentrum einer Instanz der Kernfunktion, die einen Einflußbereich der Breite h besitzt.

69.

Interpretation:

- ☐ Interpretierer wertet Anfrage direkt zur Laufzeit aus
 - langsamere Ausführung durch
 - a) "schlechtere" Optimierung
 - b) Zugriff auf die Metadaten (Kataloginformation) zur Laufzeit
 - c) AWP mit ähnlichen Anfragen in einer Schleife
 - schnellere Ausführung durch
 - a) Berücksichtigung des aktuellen DB-Zustands
- ☐ interessant für Ad-Hoc-Anfragen bzw. dynamische SQL-Anweisungen (PREPARE / EXECUTE)



71.

Zusammenfassung

- ❑ Anfrageoptimierung ist eine der schwierigsten Aufgaben in einem Datenbanksystem
- ❑ Forschungsarbeiten werden derzeit in folgenden Gebieten geleistet
 - Anfrageoptimierung in OODBMS und Datawarehousing
 - Regelbasierte Anfrageoptimierung
 - Kostenmodelle
 - Statistische Methoden