UNIVERSITÄT DES SAARLANDES
Fachbereich 14 – Informatik
Prof. Dr.-Ing. G. Weikum
Dipl.-Inform. R. Schenkel

Datenbanksysteme WS 1999/2000

9. Übung Abgabe: 17.01.2000 (in der Vorlesung)

Aufgabe 1: Verteilte Ausführung von Joins

Eine Firma habe ein verteiltes Datenbanksystem mit den Standorten Merzig (MZG), Homburg (HOM) und Saarlouis (SLS). Gegeben ist die folgende Query in Merzig:

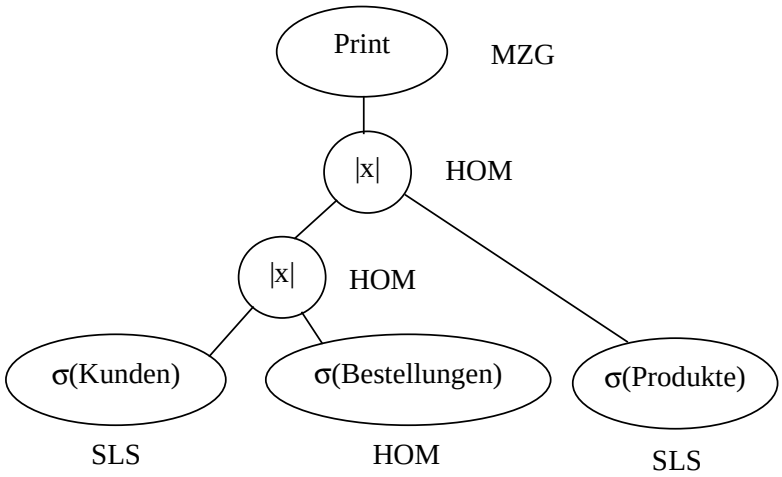
Π[PNr,Bez,KNr,Name,Gewicht,Menge]
(σ[Ort="Bexbach"](Kunden) |x| σ[Monat=10](Bestellungen) |x| σ[Gewicht>100](Produkte))

Es sollen die folgenden Annahmen über Kardinalitäten und Attributlängen gelten:

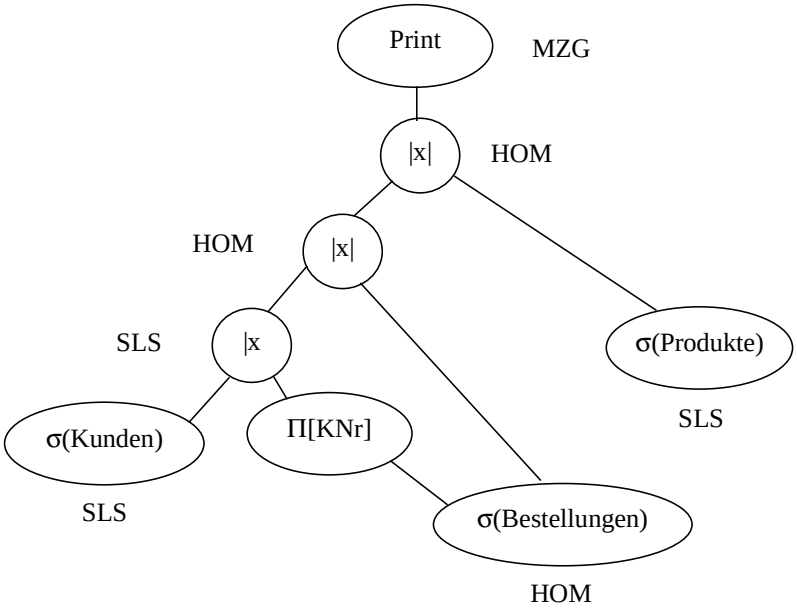
card(σ[Ort="Bexbach"](Kunden)) = 4000 Tupel
card(σ[Monat=10](Bestellungen)) = 20000 Tupel
card(σ[Gewicht>100](Produkte)) = 1000 Tupel
length(Bez)=length(Name)= 50 Bytes
length(PNr)=length(KNr)=length(Gewicht)=length(Menge) = 4 Bytes

Werteverteilungen verschiedener Attribute seien unabhängig voneinander und jeweils gleichverteilt. Die Relationen Kunden und Produkte seien in Saarlouis gespeichert, und die Relation Bestellungen sei in Homburg gespeichert.

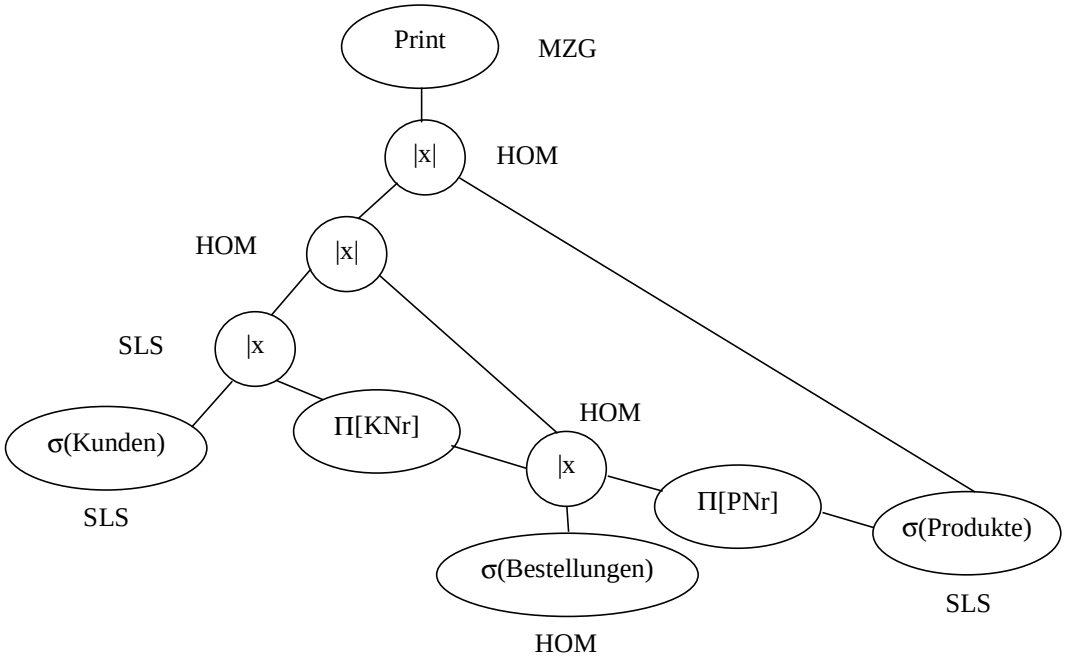
Der Query-Optimierer habe als Join-Reihenfolge festgelegt, daß zunächst *σ(Kunden)* |x| *σ(Bestellungen)* berechnet werden soll, und zwar in Homburg, und der Join mit *σ(Produkte)* anschließend in Homburg erfolgen soll. Es ergibt sich also folgender Ausführungsplan:



Schätzen Sie die notwendigen Kommunikationskosten (in Anzahl zu versendender Bytes)
a) für diesen Ausführungsplan,
b) für den folgenden um einen Semijoin (|x) erweiterten Ausführungsplan



c) für den folgenden um ein Semijoin-Programm erweiterten Ausführungsplan



Aufgabe 2: Lastbalancierung bei parallelen Hash-Joins

Bei einem parallelen Hash-Join können Lastunbalanciertheiten (engl.: Skew) auf den Join-Knoten auftreten, die den erzielten Speedup gegenüber dem theoretisch erreichbaren Speedup stark reduzieren können.

Welche Möglichkeiten zur verbesserten Lastbalancierung sehen Sie

a) vor Beginn der eigentlichen Join-Operation, also z.B. beim Versenden von Daten an die Join-Knoten, und
b) während der Join-Berechnung, z.B. durch Austauschen von Daten zwischen Join-Knoten.

Bei beiden Varianten kann die Organisation der Daten in Hash-Buckets ausgenutzt werden.