

Query Optimisation Tutorial

SWEN304/SWEN439

Lecturer: Dr Hui Ma

Engineering and Computer Science



Query Computation Costs for Unary Operators

- selection σ_C is linear in the size number n of tuples of the involved relation
 - scan the relation one tuple after the other
 - check for each tuple, whether the condition C is satisfied or not
 - keep exactly those tuples satisfying C
- projection π_{AL} is in $O(n \cdot \log n)$ with the number n of tuples
 - order the relation according to the attributes in AL
 (this is the most costly part leading to the complexity in $O(n \cdot \log n)$)
 - scan the relation one tuple after the other
 - project each tuple to the attributes in AL and check, whether result is the same as for previous tuple (duplicate elimination)
 - **Note:** SQL does not eliminate tuples, i.e. costs of projection are in $O(n)$, but DISTINCT needs the ordering
- renaming δ_f can be neglected

Query Computation Costs for Binary Operators

- join \bowtie is in $O(n \cdot \log n)$ with $n = n_1 + n_2$, where n_i are the respective numbers of tuples in the two relations involved
 - the easiest idea is to use a **nested loop**:
 - scan the first relation one tuple after the other
 - for each tuple scan the second relation to find matching tuples, i.e., those coinciding with the given tuple on the common attributes
 - in case tuples match, take the joined tuple into the result relation
 - more efficient is the **merge join**:
 - sort both relations (this is the most costly part)
 - scan both relations simultaneously to find matching tuples
 - in case tuples match, take the joined tuple into the result relation
- union \cup is in $O(n \cdot \log n)$ with $n = n_1 + n_2$, where n_i are the respective numbers of tuples in the two relations involved (analogously for difference $-$)
 - sort both relations as for the merge join
 - scan simultaneously to detect duplicates

Estimating the Size of Relations

- let $R = \{A_1, \dots, A_k\}$ be a relation schema
- determine the size of a relation r over R :
 - let n denote the average number of tuples in the relation r
 - let ℓ_j denote the the average space (e.g., in bits) for attribute A_j in a tuple in r
 - then $n \cdot \sum_{j=1}^k \ell_j$ is the space needed for the relation r
- determine the size of intermediate relations in a query using the query tree:
 - assign the size of the relation to each leaf node R
 - for a renaming node the assigned size is exactly the size s assigned to the successor

Estimating the Size of Intermediate Results

- for a selection node σ_C the assigned size is $a_C \cdot s$, where s is the size assigned to the successor and $100 \cdot a_C$ is the average percentage of tuples satisfying C
- for a projection node π_{R_i} the assigned size is $(1 - C_i) \cdot s \cdot \frac{r_i}{r}$, where r_i (r) is the average size of a tuple in a relation over R_i (R), s is the size assigned to the successor and C_i is the probability that two tuples coincide on R_i

$$(1 - C_i) \cdot s \cdot \frac{r_i}{r} = (1 - C_i) \cdot n \cdot r_i$$
where n is average number of tuples in R -relation
- for a join node the assigned size is $\frac{s_1}{r_1} \cdot p \cdot \frac{s_2}{r_2} \cdot (r_1 + r_2 - r)$, where s_i are the sizes of the successors, r_i are the corresponding tuple sizes, r is the size of a tuple over the common attributes and p is the matching probability
- for a union node the assigned size is $s_1 + s_2 - p \cdot s_1$ with the probability p for tuple of R_1 to coincide with a tuple over R_2
- for a difference node the assigned size is $s_1 \cdot (1 - p)$ where $(1 - p)$ is probability that tuple from R_1 -relation does not occur as tuple in R_2 -relation

Natural join needs to remove duplicate attributes
For equi-join, $r = 0$

Estimating the Size of Intermediate Results

- Person = {Name, Age, Address} with minimal key {Name, Address}
- Customer = {CustNo, CustName, CustAddress} with minimal key {CustNo} and foreign key [CustName, CustAddress] \subseteq Person[Name, Address]

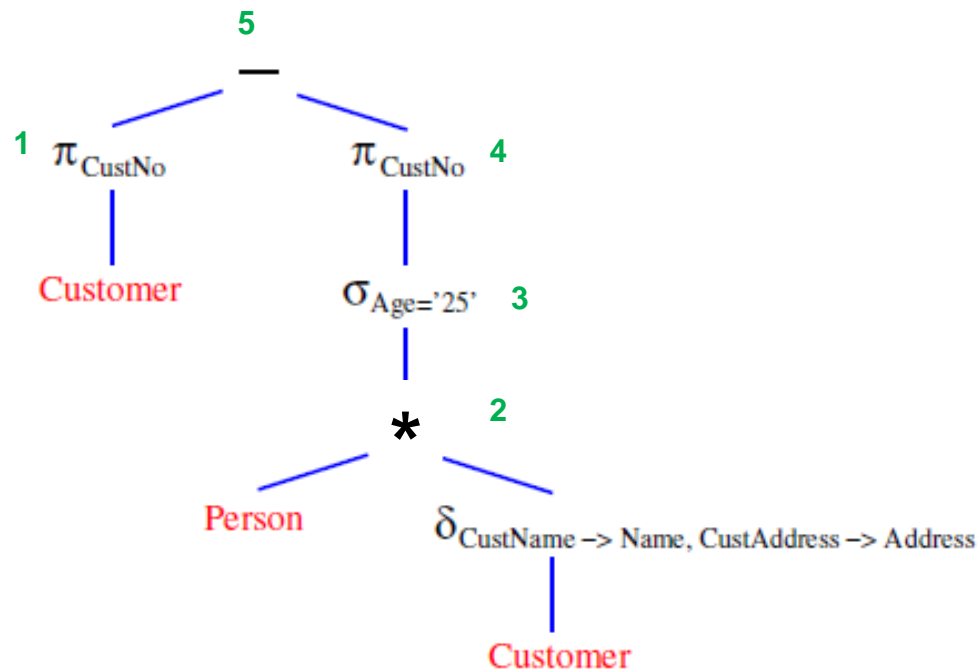
| attribute | domain | average length |
|-------------|-------------|----------------|
| Name | VARCHAR(30) | 15 |
| Address | VARCHAR(50) | 30 |
| CustName | VARCHAR(30) | 15 |
| CustAddress | VARCHAR(50) | 30 |

Assumptions

- Assume that the fixed number of bits for storing the age of a person is 8,
 - For values up to $2^8=256$
- assume to have 1000 customers in our database, and 1010 different people
- assume that there are exactly 5% of customers aged '25', the value for a_c is 0.05,

Estimating the Size of Intermediate Results

- $\pi_{\text{CustNo}}(\text{Customer}) - \pi_{\text{CustNo}}(\sigma_{\text{Age} = '25'}(\text{Person} * \delta_{\text{CustName} \rightarrow \text{Name}, \text{CustAddress} \rightarrow \text{Address}}(\text{Customer})))$



Estimating the Size of Intermediate Results

- Compute the size of tuple of **Customer**

$$r_{customer} = 15 \cdot 8 + 10 + 30 \cdot 8 = 370 \text{ bits}$$

- *Note:* we need 10 bits to store the customer number, if there are 1,000 customers ($2^{10} = 1,024$)
- Average size of relation Customer

$$S_{customer} = 1,000 \cdot 370 = 370,000 \text{ bits}$$

- Computer the size of tuple **Person**

$$r_{person} = 15 \cdot 8 + 8 + 30 \cdot 8 = 368 \text{ bits}$$

- Average size of a relation over Person:

$$S_{person} = 1,010 \cdot 368 = 371,680 \text{ bits}$$

Estimating the Size of Intermediate Results

- For the join node (**node 2**), the probability $p = \frac{1}{1010}$
- The attributes of the relation resulted from the join are:
- {Name, Address, Age, Customer}

$$r_2 = r_{\text{Name}} + r_{\text{Address}} + r_{\text{Age}} + r_{\text{Customer}}$$

$$= 15 \cdot 8 + 30 \cdot 8 + 8 + 10 = 378 \text{ bits}$$

- Average size of the relation of the join node

$$S_2 = \frac{s_{\text{person}}}{r_{\text{person}}} \cdot \frac{1}{1010} \cdot \frac{s_{\text{customer}}}{r_{\text{customer}}} r_2 = 1000 \cdot 378 = 378,000 \text{ bits}$$

- For selection node $\sigma_{\text{Age} = '25'}$ (**node 3**), $a_c = 5\%$

$$S_3 = 0.05 \cdot 378,000 = 18,900 \text{ bits}$$

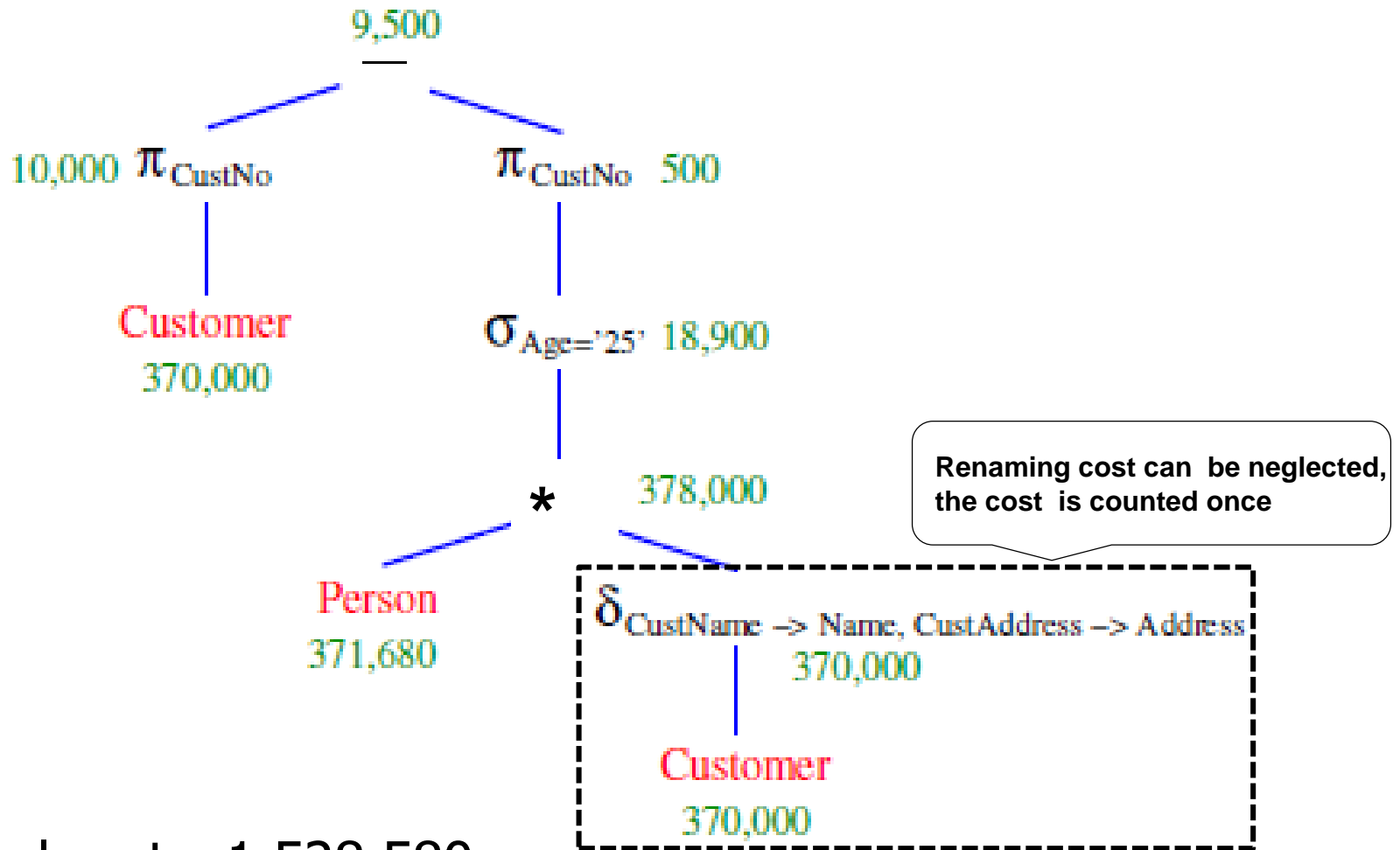
- For project node π_{CustNo} (**node 4**), $C = 0\%$

$$s_4 = 18,900 \cdot 10/378 = 500 \text{ bits}$$

Estimating the Size of Intermediate Results

- For the projection π_{CustNo} (**node 1**), $C = 0\%$
 $s_1 = 370,000 \cdot 10/370 = 10,000$ bits
- For the difference (**node 5**), $p = 5\%$
 $s_5 = 10,000 \cdot (1 - 0.05) = 10,000 \cdot 0.95 = 9,500$ bits

Estimating the Size of Intermediate Results



- Total cost = 1,528,580

Cost Related Catalog Content

- For the purpose of a query cost estimate, a Catalog should contain following information for each **base relation**:
 - Number of tuples (= records) n
 - Number of blocks b
 - Blocking factor f (= the number of tuples that fit into one block)
 - Available access methods and access attributes:
 - Access methods: sequential, indexed, hashed
 - Access attributes: primary key, indexing attributes,
 - The number of levels h of each index
 - The number of distinct values d of each attribute

Cost Functions of Select Operation

■ Remark:

- **Linear search** (neither indexes nor hash functions provided)

$$C = b + \lceil s / f \rceil, \text{ hence } O(n)$$

- **Unique key index** (B^+ -tree):

- If selection condition is $K = k$:

$$C = h + 1 + \lceil 1 / f \rceil$$

Hence $O(\log n)$ – index height h is proportional to $\log r$

- If selection condition is $k_1 \leq K \leq k_2$ and suppose $s \leq n$ tuples satisfy the condition:

Hence $O(\max\{\log n, s\})$

Cost Functions of Select Operation

- **Secondary index** (B^+ -tree) on secondary key Y
 - $s \leq d(Y)$ random tuples satisfy condition $Y = y$
 - each Y value has a pointer to a sequence of blocks containing up to p pointers to tuples in the data area
 - the height h of the tree is proportional to $\log(d(Y))$

$$C = h + \lceil s / p \rceil + s + \lceil s / f \rceil,$$

- Hence $O(s)$

Exercise

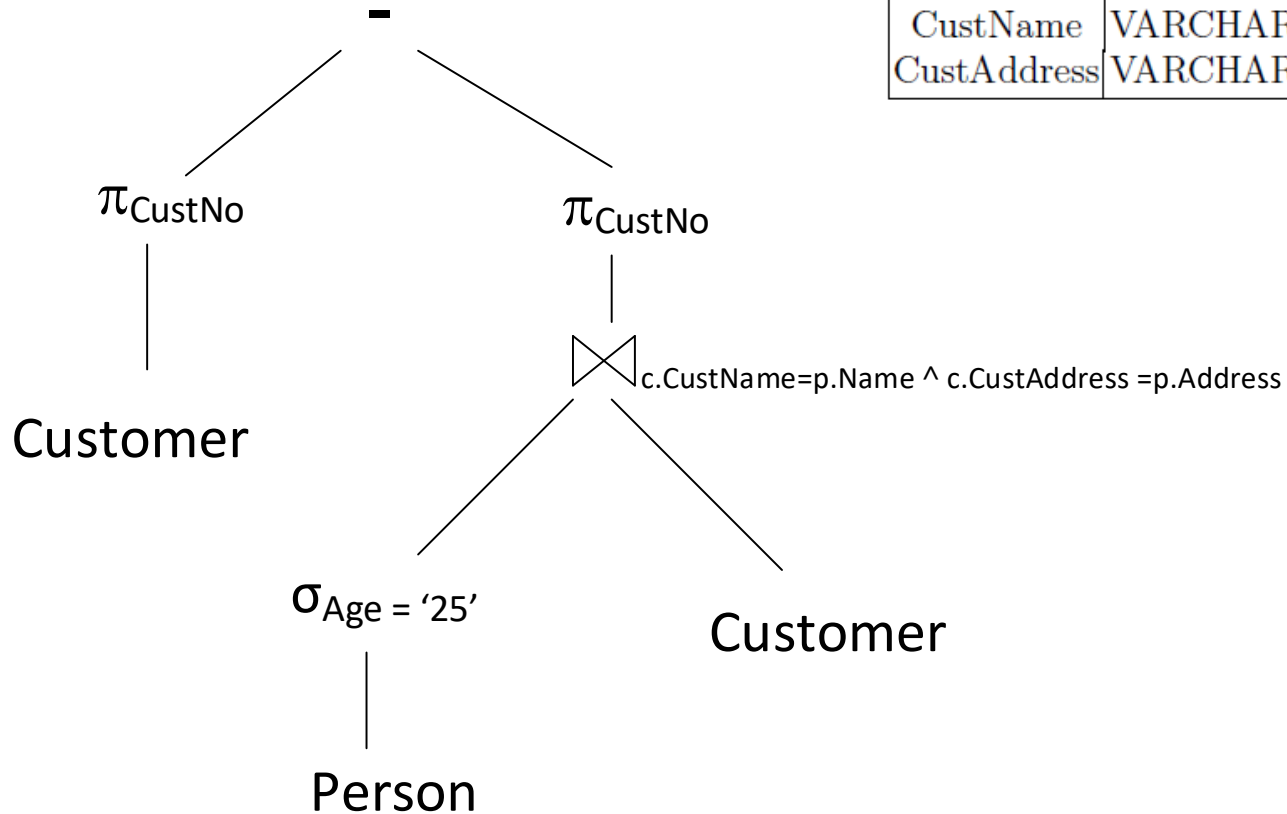
■ Compute the total cost of the following query tree

- Person = {Name, Age, Address} with minimal key {Name, Address}
- Customer = {CustNo, CustName, CustAddress} with minimal key {CustNo} and foreign key [CustName, CustAddress] \subseteq Person[Name, Address]

$$r_{customer} = 15 \cdot 8 + 10 + 30 \cdot 8 = 370 \text{ bits}$$

$$r_{person} = 15 \cdot 8 + 8 + 30 \cdot 8 = 368 \text{ bits}$$

| attribute | domain | average length |
|-------------|-------------|----------------|
| Name | VARCHAR(30) | 15 |
| Address | VARCHAR(50) | 30 |
| CustName | VARCHAR(30) | 15 |
| CustAddress | VARCHAR(50) | 30 |



Exercise

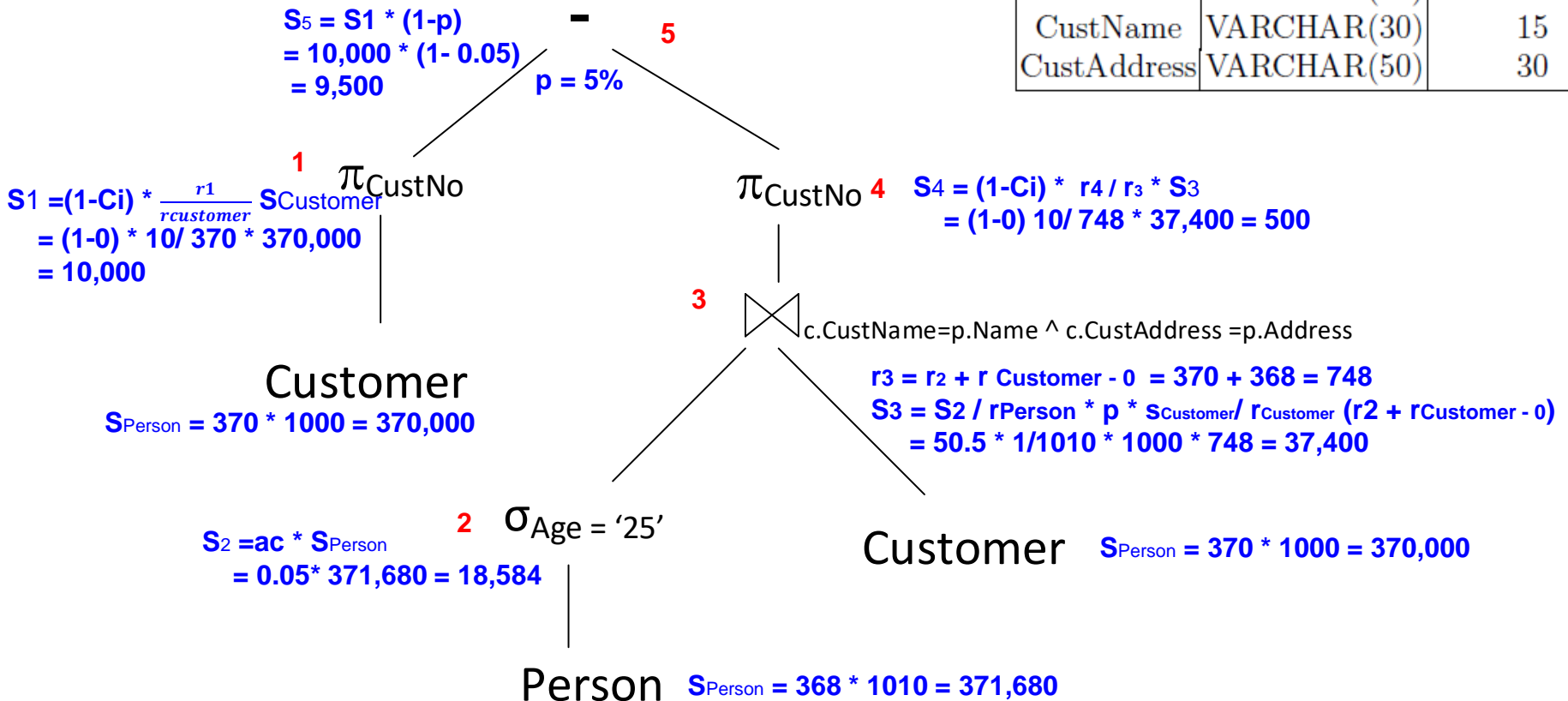
■ Compute the total cost of the following query tree

- Person = {Name, Age, Address} with minimal key {Name, Address}
- Customer = {CustNo, CustName, CustAddress} with minimal key {CustNo} and foreign key [CustName, CustAddress] \subseteq Person[Name, Address]

$$r_{customer} = 15 \cdot 8 + 10 + 30 \cdot 8 = 370 \text{ bits}$$

$$r_{person} = 15 \cdot 8 + 8 + 30 \cdot 8 = 368 \text{ bits}$$

| attribute | domain | average length |
|-------------|-------------|----------------|
| Name | VARCHAR(30) | 15 |
| Address | VARCHAR(50) | 30 |
| CustName | VARCHAR(30) | 15 |
| CustAddress | VARCHAR(50) | 30 |



Total = 1,187,664