## Chapter 4.3 Nested Loop Join

We need to pay attention to *Join Operator Algorithm* which is named as *Nested Loop Join*, the cost of Nested Loop Join is *one pass with half*. Among all kinds of algorithms, tuples of one Relation will be read for once while tuples of another Relation will be read repeatly.

Nested Loop Join can be used on Relation with any size; There is no need to store the whole Relation into the main memory.

### Chapter 4.3.1 Nested Loop Join based on Tuple

***Introduction:***

The simplest loop is aim to each tuple of Relation. For *Nested Loop Join based on Tuple* algorithm, we connect *R(X, Y) AND S(Y, Z)*:

*For each tuple s in Relation S DO:*

*For each tuple r in Relation R DO:*

*IF r connects with s and form the tuple t then:*

*output tuple t;*

***Cost:***

If we do not pay attention on the block buffer method in Relation R and Relation S, then the cost of I/O will reach up to *T(R)T(S)*.

***Improvement:***

1. Using the index of join property to find the specific r tuple for s tuple, do not need to read the whole Relation R.
2. Pay more attention on How tuple r and s are distributed in the main memory. Then when we execute the inner loop, then use the main memory as much as we can to make the disk I/O less and less.

### Chapter 4.3.2 Nested Loop Join Iterator based on Tuple

### Chapter 4.3.3 Nested Loop Join Algorithm based on Block

### Chapter 4.3.4 Nested Loop Join Analysis