## Chapter 5.3 From Grammar Analysis Tree to Logic Query Plan

In the Chapter 5.1, we have already construct a Grammar Analysis Tree of Query Statement, then in the next step, we need to convert the Grammar Tree to the Logic Query Plan.

***Steps:***

1. Using one or more Relation Algebra Operator to substitute node and structure in the Grammar Tree according to appropriate Group.
2. Convert the Relation Algebra Operator to the expected Expression, here it may be converted to the most efficient Physical Query Plan.

### Chapter 5.3.1 Convert to Relation Algebra

Here we may explain some rules to convert SQL Grammar Tree to the Algebra Logic Query Plan.

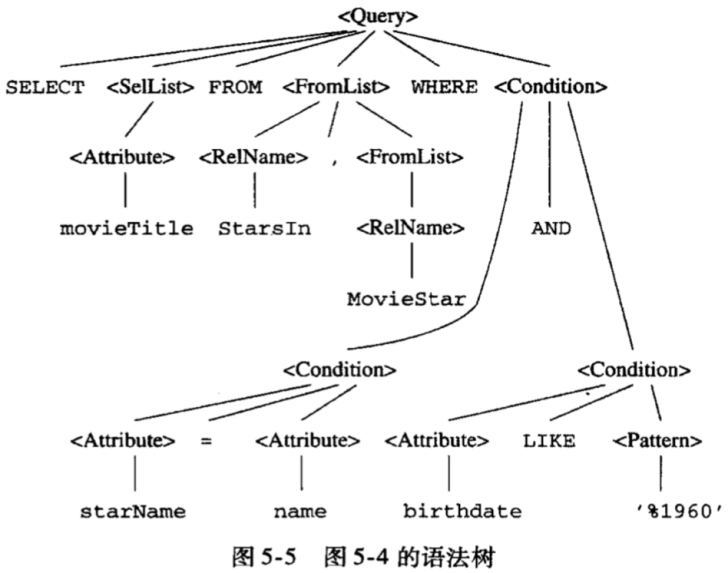
***Rule:****(This rule makes us to convert the simple ‘select - from - where’ structure to the Relation Algebra.)*

* If there has one <Condition> without sub <Query>, then we can use one Relation Algebra Expression to substitute the whole part - Selection List, from list and condition, the Algebra Expression from bottom to top consist with the contents below:

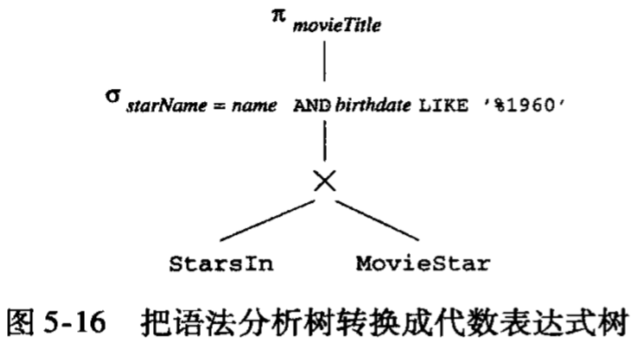
1. <FromList> - all Relation Product is the Operator Parameter for below.
2. Selection C, here C needs to be substituted by the <Condition > Expression, also Selection is the Operator Parameter for below.
3. Projection L, here L is the attributes list in the <SelList>.

***Example:***

Let’s consider the Grammar Analysis Tree in the example 5 - 5.



Here, we get the ***(1)*** Product of two Relations StarsIn and MovieStar in the ***from list***, and ***(2)*** proceed Selection by using the sub - tree in ***<Condition> root***, and ***(3)*** Project to the Selection list movieTitle. Finally, get the Relation Algebra Expression below:



### Chapter 5.3.2 Remove Sub - Query from Condition

***Principle:***

For <Condition> that includes the Sub - Query Grammar Tree, we introduce the intermediate operator format, it is between Grammar Type of Grammar Analysis Tree and Relation Algebra Operator that operates on Relation. It is called *Two Parameter Selection*.

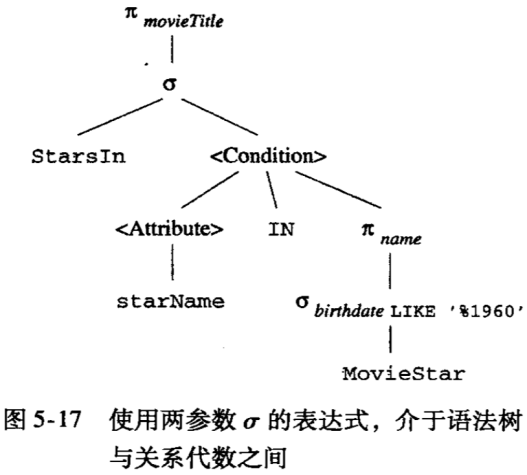
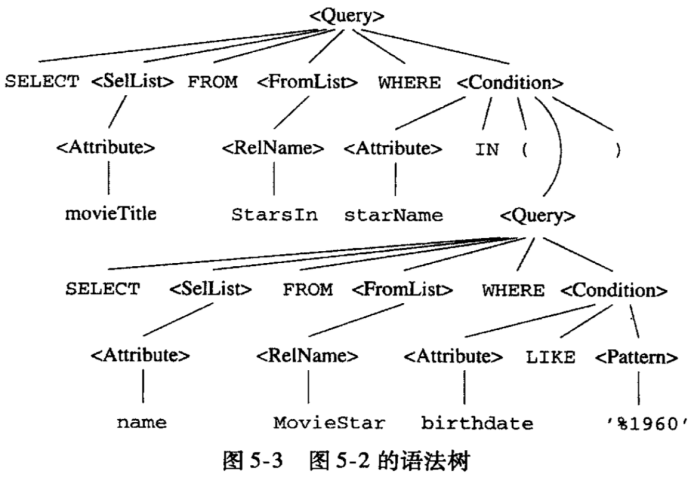
Here we use the node with the *tag α* and without any parameters to present the *Two Parameters Selection after conversion*. Also there exists one left node and it presents the Relation R which we need to operate on it, and also one right node, it presents the Condition Expression which has role on the Relation R.

*(Attention that two parameters can be presented as Grammar Tree, Expression Tree or combination of both.)*

***Example:***

Picture 5 - 17 is rewrite of Picture 5 - 3 by using Two Parameters Selection   
Operator for the Grammar Tree. There exist several conversions when constructing the new Grammar Tree.

1. The Sub - Query is substituted by one Relation Algebra Expression in 5 - 17.
2. Substituted by using *select - from - where* in outer query. Of course, here we use the *Two Parameter Selection* to present the necessary Selection, but not the common Relation Algebra Operator. Also, the upper Grammar Level Node <Condition> has not been substituted, but instead it still be as one parameter in Selection, and its round bracket and <Query> has been substituted by the Relation Algebra in the first point.



***Improvement:***

However, the tree above needs the further conversion, and the rule is needed to use the Single Parameter Operator and other Relation Operator to substitute the Two Parameters Selection Operator.

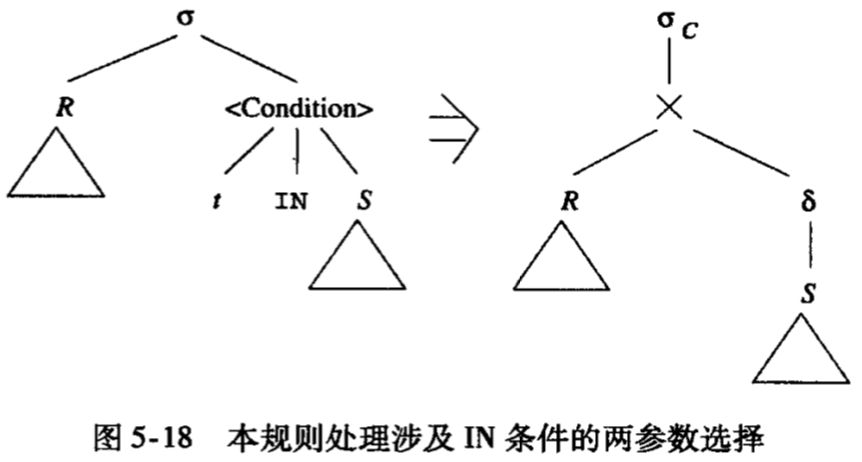
As an example, we will give the rule which relates with IN Operator. Attention here, the Query in the Sub - Query is not related, which is to say this Sub - Query can only be calculated once, since it has no relation with the checked tuples.

Here is the rule to eliminate such condition, and assume that we have a *Two Parameters Selection:*

1. One of the parameter represents the some Relation R.
2. The second parameter looks like *<condition> t IN S.*
3. Expression S is a Non - Relation Query.
4. t is a tuple consist by some attributes from Relation R.

***Conversion:***

1. Using the Expression Tree of S to substitute <Condition>. If S has repetition, then it is necessary to keep a Two Parameter Selection Operator in the root of S Expression, therefore the formed tuples will not exceed the number of tuples from original Query.
2. Using the Single Selection Operator to substitute the Two Parameter Selection, here we need to pay attention that condition C is each field in tuple t gets equal value with the corresponding field in Relation S.
3. Here give another parameter to Single Selection Operator, it is the product of R and S.



***Example:***

***Example:***

### Chapter 5.3.3 Improvement from Logic Query Plan

### Chapter 5.3.4 Grouping of Combinative and Distributable Operator