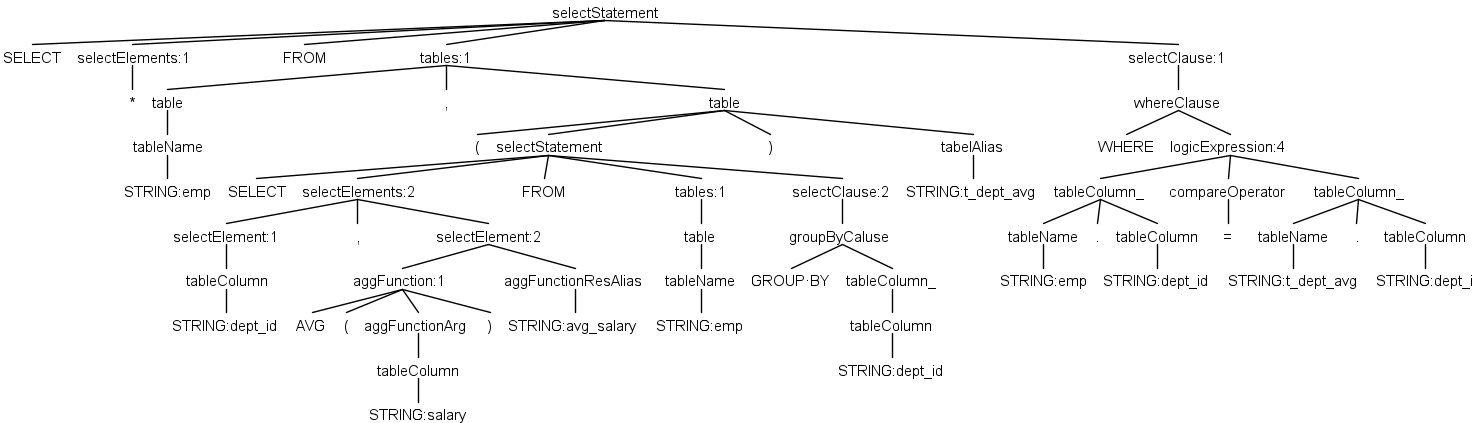
作业1：Antrl4构建sql查询语法分析树

查询语句：

1. **SELECT** \* **FROM** emp,(
2. **SELECT** dept\_id , AVG(salary) avg\_salary
3. **FROM** emp
4. **GROUP** **BY** dept\_id
5. ) t\_dept\_avg
6. **WHERE** emp.dept\_id = t\_dept\_avg.dept\_i

生成的语法分析树：



作业2：构建真实的语言类应用程序

对于sql查询语句：

1. #示例1
2. **SELECT** id,score **FROM** students **ORDER** **BY** score LIMIT 3 OFFSET 3
3. #示例2
4. **SELECT** last\_name, job\_id, salary
5. **FROM** employees
6. **WHERE** job\_id=(
7. **SELECT** job\_id
8. **FROM** employees
9. **WHERE** employee\_id = 141 AND last\_name = ''Abel''
10. )
11. #示例3
12. **SELECT** last\_name, e.salary
13. **FROM** departments d **INNER** JOIN employees e
14. **ON** d.department\_id = e.department\_id
15. **WHERE** d.location\_id IN (1400, 1700)

根据sql查询语句逆向推测数据库表结构。

|  |  |
| --- | --- |
| **Table:**employees | |
| job\_id |  |
| last\_name | VARCHAR(20) |
| salary |  |
| employee\_id | INT |

|  |  |
| --- | --- |
| **Table:**students | |
| id |  |
| score |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Table:**employees | | **Table:**departments | |
| department\_id |  | department\_id |  |
| last\_name? |  | location\_id | INT |
| salary |  | last\_name? |  |

运行结果：







根据sql查询语句自动构造测试数据

作业3：go语言编译器

三地址码可以看成是语法树或DAG的线性表示。

[{result=t0, op=\*, arg2=2, arg1=b},

{result=t1, op=+, arg2=t0, arg1=a},

{result=c, op==, arg1=t1}]

c = a+b\*2

