



Like and Wildcards Refactoring

Marcos Irazabal - Maria Rosa Marcheschi



Like and Wildcards Refactoring

Definición: El uso de operadores Like con wildcards (%) puede causar problemas de rendimiento.

Es mejor usar % al final de las sentencias, en lugar de realizar una exploración completa de la tabla

```
"SELECT * FROM table_name  
WHERE nombre LIKE 'a%r%';"
```



```
"SELECT * FROM table_name  
WHERE nombre LIKE 'ar%';"
```





Like and Wildcards Refactoring

- Precondiciones (entrada)
 - Aplicable unicamente a “consultas” sintacticamente correctas (parsea ok)
 - La consulta a refactorizar contiene uno o múltiples “%” pero este no esta solo posicionado al final de la expresión del LIKE → para esto usamos un visitor que chequea la cantidad y ubicación del/los wildcard/s.
- Post Condiciones (salida)
 - Es una consulta válida (parsea ok)
 - Hay un solo “%” al final de la expresión LIKE → para esto usamos otro visitor que comprueba esto



Problemas a los que nos enfrentamos

- Precondiciones, cómo evaluar donde hay un “ % ”  Leímos la documentación del SQLite para descubrir que no puede haber un “%” fuera de la sentencia LIKE
- Como modificar el nodo terminal LIKE sin sobrescribir toda la consulta a refactorizar  redefiniendo los métodos adecuados del “SQLiteParserBaseVisitor”.



Implementación

- LikeVisitor
 - Re-implementamos el método VisitExpr y verificamos estemos en un nodo del tipo LIKE
 - Esto lo hacemos con “CTX.LIKE_()”
 - Dentro de este método modificamos el valor del texto del LIKE para eliminar todos los % y agregar uno solo al final
- Conclusiones
 - El trabajo es básicamente realizar un template method (método transform) usando un patron visitor para ir recorriendo el árbol e ir realizando acciones en distintos tipos de nodos terminales
 - No hizo falta realizar un visitor para chequear si el % estaba solamente en el LIKE porque es el único lugar donde puede aparecer