

## 1.1 ./csharp/Platform.RegularExpressions.Transformer.HasuraSQLSimplifier/HasuraSQLSimplifierTransformer.cs

```

1  using System.Collections.Generic;
2  using System.Linq;
3  using System.Text.RegularExpressions;
4
5  #pragma warning disable CS1591 // Missing XML comment for publicly visible type or member
6
7  namespace Platform.RegularExpressions.Transformer.HasuraSQLSimplifier
8  {
9      public class HasuraSQLSimplifierTransformer : TextTransformer
10     {
11         public static readonly IList<ISubstitutionRule> DefaultRules = new List<SubstitutionRule>
12         {
13             // HTML clean up
14             (new Regex(@"<span class=""[^"""]*">([<>]*)</span>"), "$1", 0),
15             // ('describe')
16             // 'describe'
17             (new Regex(@"\[\\s\\n]*('[']+')[\\s\\n]*\\s\\n)"), "$1", int.MaxValue),
18             // AND ('true' AND 'true')
19             //
20             (new Regex(@"\[\\s\\n]*AND\[\\s\\n]*\[\\s\\n]*'true' \[\\s\\n]*AND\[\\s\\n]*'true' \[\\s\\n]*\\s\\n)"), "",
21             ↪ 0),
22             // AND ('true')
23             //
24             (new Regex(@"\[\\s\\n]*AND\[\\s\\n]*'true'"), "", 0),
25             // ::
26             // ::
27             (new Regex(@"\[\\s\\n]*::\[\\s\\n]*"), "::", 0),
28             // ('describe'::text)
29             // 'describe'::text
30             (new Regex(@"\[\\s\\n]*('[']+)::text \[\\s\\n]*\\s\\n)"), "$1", 0),
31             // ("_0__be_0_nodes"."target_id")
32             // "_0__be_0_nodes"."target_id"
33             (new Regex(@"\[\\s\\n]*(""[^"""]*"") \[\\s\\n]*\\. \[\\s\\n]*(""[^"""]*"") \[\\s\\n]*\\s\\n)"), "$1.$2",
34             ↪ 0),
35             // ("public"."nodes"."_id")
36             // "public"."nodes"."_id"
37             (new Regex(@"\[\\s\\n]*(""[^"""]*"") \[\\s\\n]*\\. \[\\s\\n]*(""[^"""]*"") \[\\s\\n]*\\. \[\\s\\n]*(""[^"""]*"") \[\\s\\n]*\\s\\n)"), "$1.$2.$3",
38             ↪ 0),
39             // LIMIT\\n\\t\\t\\t1
40             // LIMIT 1
41             (new Regex(@"(LIMIT) \[\\s\\n]*\\s\\n(\d+)"), "$1 $2", 0),
42             // ("_0__be_0_nodes"."type" = 'describe'::text)
43             // "_0__be_0_nodes"."type" = 'describe'::text
44             (new Regex(@"(\\W) \[\\s\\n]*(((?!SELECT) [^()])*) \[\\s\\n]*\\s\\n)"), "$1$2", int.MaxValue),
45         }.Cast<ISubstitutionRule>().ToList();
46
47         public HasuraSQLSimplifierTransformer()
48         : base(DefaultRules)
49     {
50     }
51 }

```

## 1.2 ./csharp/Platform.RegularExpressions.Transformer.HasuraSQLSimplifier.Tests/HasuraSQLSimplifierTransformerTests.cs

```

1  using Xunit;
2
3  namespace Platform.RegularExpressions.Transformer.HasuraSQLSimplifier.Tests
4  {
5      public class HasuraSQLSimplifierTransformerTests
6      {
7          [Fact]
8          public void EmptyLineTest()
9          {
10             // This test can help to test basic problems with regular expressions like incorrect
11             ↪ syntax
12             var transformer = new HasuraSQLSimplifierTransformer();
13             var actualResult = transformer.Transform("");
14             Assert.Equal("", actualResult);
15
16             [Fact]
17             public void BasicRequestTest()
18             {
19                 var original = @"SELECT
20 coalesce(json_agg("root"), '[]') AS "root"
21 FROM

```

```

(
  SELECT
    row_to_json(
      (
        SELECT
          ""_2_e""
        FROM
          (
            SELECT
              ""_1_root.base"".""id"" AS ""id""
            ) AS ""_2_e""
          )
        ) AS ""root""
    FROM
      (
        SELECT
          *
        FROM
          ""public"".""nodes""
        WHERE
          (
            (
              (""public"".""nodes"".""type"" = (('auth_token') :: text))
            )
            AND (
              EXISTS (
                SELECT
                  1
                FROM
                  ""public"".""nodes"" AS ""_0__be_0_nodes""
                WHERE
                  (
                    (
                      (
                        (""_0__be_0_nodes"".""source_id"" = (""public"".""nodes"".""id""))
                      )
                      AND ('true')
                    )
                    AND (
                      (
                        (
                          (""_0__be_0_nodes"".""type"" = (('describe') :: text))
                          AND ('true')
                        )
                        AND (
                          (
                            (
                              (""_0__be_0_nodes"".""target_id"" = (('X-Hasura-User-Id') :: text))
                            )
                            AND ('true')
                          )
                          AND ('true')
                        )
                      )
                    )
                    AND (
                      ('true')
                      AND ('true')
                    )
                  )
                )
              )
            )
          )
        ) AS ""_1_root.base""
      )
    LIMIT
      1
    ) AS ""_3_root"";

    var expected = @"SELECT
  coalesce(json_agg(""root""), '[]') AS ""root""
FROM
  (
    SELECT
      row_to_json(
        (
          SELECT
            ""_2_e""
          FROM
            (
              SELECT

```

```

102         ""_1_root.base"".""id"" AS ""id""
103     ) AS ""_2_e""
104 )
105 ) AS ""root""
106 FROM
107 (
108     SELECT
109     *
110     FROM
111     ""public"".""nodes""
112     WHERE
113     (
114         ""public"".""nodes"".""type"" = 'auth_token'::text
115         AND (
116             EXISTS (
117                 SELECT
118                 1
119                 FROM
120                 ""public"".""nodes"" AS ""_0__be_0_nodes""
121                 WHERE
122                 ""_0__be_0_nodes"".""_source_id"" = ""public"".""nodes"".""_id""
123                 AND ""_0__be_0_nodes"".""type"" = 'describe'::text
124                 AND ""_0__be_0_nodes"".""target_id"" = 'X-Hasura-User-Id'::text
125             )
126         )
127     )
128 ) AS ""_1_root.base""
129 LIMIT 1
130 ) AS ""_3_root"";
131     var transformer = new HasuraSQLSimplifierTransformer();
132     var actual = transformer.Transform(original);
133     Assert.Equal(expected, actual);
134 }
135 }
136 }

```

## Index

./csharp/Platform.RegularExpressions.Transformer.HasuraSQLSimplifier.Tests/HasuraSQLSimplifierTransformerTests.cs, 1  
./csharp/Platform.RegularExpressions.Transformer.HasuraSQLSimplifier/HasuraSQLSimplifierTransformer.cs, 1