

1.-

let \$x := //asignatura/num_horas

return avg(\$x)

The screenshot shows the BaseX 9.7.2 interface. The query editor contains the following XQuery:

```

let $x := //asignatura/num_horas
return avg($x)

```

The result pane shows a single result: 4.2. The visualizer displays a tree structure of the XML document 'Listado.xml'. The tree has a root element 'Listado' with three children: 'asignaturas', 'profesores', and 'cursos'. The 'asignaturas' element has five children, each with a 'num_horas' attribute. The 'profesores' element has eight children, each with a 'nombre' attribute. The 'cursos' element has eight children, each with a 'nombre' attribute. The result 4.2 is the average of the 'num_horas' values.

2-

for \$x in doc("Listado")//asignatura

where \$x/@prof=//profesor[nombre='Antonio']/@id

return data(\$x/nombre)

The screenshot shows the BaseX 9.7.2 interface. The query editor contains the following XQuery:

```

for $x in doc("Listado")//asignatura
where $x/@prof=//profesor[nombre='Antonio']/@id
return data($x/nombre)

```

The result pane shows two results: 'Historia' and 'Física y Química'. The visualizer displays the same tree structure as the first screenshot. The result is the list of 'nombre' values for the 'asignatura' elements where the 'prof' attribute matches the 'id' attribute of the 'profesor' element with the name 'Antonio'.

```
return data($x/nombre)
```

```
return distinct-values(concat('Nivel: ', $x/nivel, ' Grupo: ', $x/grupo))
```

[illegible]

5-

let \$x :=//asignatura[2]/nombre

return string-length(\$x)

The screenshot shows the BaseX 9.7.2 interface. The query editor contains the following XQuery:

```
1 let $x :=//asignatura[2]/nombre
2 return string-length($x)
```

The result pane shows a single result: 16. The XML tree view on the right shows the structure of the 'Listado.xml' file, with the path //asignatura[2]/nombre highlighted.

6-

for \$x in doc('Listado')//asignatura

where \$x/@prof=//profesor[nombre='Cristina']/@id and \$x/@curso=//curso[nivel='4º ESO' and grupo='A']/@id

return data(\$x/nombre)

The screenshot shows the BaseX 9.7.2 interface. The query editor contains the following XQuery:

```
1 for $x in doc('Listado')//asignatura
2 where $x/@prof=//profesor[nombre='Cristina']/@id and $x/@curso=//curso[nivel='4º ESO' and grupo='A']/@id
3 return data($x/nombre)
```

The result pane shows a single result: Biología. The XML tree view on the right shows the structure of the 'Listado.xml' file, with the path //asignatura[2]/nombre highlighted.

7-

let \$x := //curso[contains(nivel, 'BACHILLER')]/alumnos

return sum(\$x)

The screenshot shows the BaseX 8.7.2 interface. The query editor contains the following XQuery:

```
1 let $x := //curso[contains(nivel, 'BACHILLER')]/alumnos
2 return sum($x)
```

The result pane shows a single result: 95. To the right, a hierarchical tree diagram visualizes the XML structure. The root is 'Listado.xml', which contains a 'listado' element. This element has three children: 'asignaturas', 'profesores', and 'cursos'. The 'asignaturas' element has five children, each labeled 'asign.' followed by a course code (AFD, CST, CHL, E4A, E4B). The 'cursos' element has eight children, each labeled with a course code (B1T, B1A, B2T, B2A, and four unlabeled ones). The 'profesores' element has a single child labeled 'n'. The 'asign.' elements have children labeled 'n' and 'horas'. The 'n' elements have children labeled 'H' and 'A'. The 'horas' element has children labeled 'H' and 'A'. The 'H' elements have children labeled '4', '6', '3', '6', '2'. The 'A' elements have children labeled 'A', '30', '4', 'B', '27', '1', 'T', '25', '1', 'A', '28', '2', 'T', '22', '2', 'A', '28'.

8-

let \$x := //curso[@id=//asignatura[nombre='Musica']/@curso]/nivel

return data(\$x)

The screenshot shows the BaseX 8.7.2 interface. The query editor contains the following XQuery:

```
1 let $x := //curso[@id=//asignatura[nombre='Musica']/@curso]/nivel
2 return data($x)
```

The result pane shows a single result: 4# ESO. To the right, a hierarchical tree diagram visualizes the XML structure. The root is 'Listado.xml', which contains a 'listado' element. This element has three children: 'asignaturas', 'profesores', and 'cursos'. The 'asignaturas' element has five children, each labeled 'asign.' followed by a course code (AFD, CST, CHL, E4A, E4B). The 'cursos' element has eight children, each labeled with a course code (B1T, B1A, B2T, B2A, and four unlabeled ones). The 'profesores' element has a single child labeled 'n'. The 'asign.' elements have children labeled 'n' and 'horas'. The 'n' elements have children labeled 'H' and 'A'. The 'horas' element has children labeled 'H' and 'A'. The 'H' elements have children labeled '4', '6', '3', '6', '2'. The 'A' elements have children labeled 'A', '30', '4', 'B', '27', '1', 'T', '25', '1', 'A', '28', '2', 'T', '22', '2', 'A', '28'.

```
return count($x)
```

```
return $x/@*
```

The screenshot shows the Neo4j GUI interface. At the top, there's a menu bar with options like Database, Editor, View, Visualization, Options, and Help. Below the menu bar, there's a toolbar with various icons. The main area is divided into several panels. On the left, there's a 'Find' panel with a search bar and a list of results. In the center, there's a 'Cypher' editor with a query input field and a 'Run' button. On the right, there's a 'Results' panel showing the output of the query. The query being executed is:

```
1 for $x in doc('Listado')//signatura
2 return $x/g*
```

 The results are displayed in a table with 10 columns and 10 rows of data. The columns are labeled with the values of the 'g' property for each 'signatura' element. The data is as follows:

g	g	g	g	g	g	g	g	g	g
4	16	3	6	2					
A	30	4	B	27	1	T	25	1	A
28	2	T	22	2	A	20			