

NAME

smacqq — query tool using SMACQ

DESCRIPTION

"Smacqq" is an extensible tool for performing queries on streams of data. Users with a familiarity of SQL will immediately be comfortable using the basic features of the system. However, there are additional object-relational and extensibility features that are described below.

The primary difference from standard databases, is that data is not stored in preloaded tables, but is instead produced by data source modules. Also, the select operation does not automatically print fields. If printable output is desired, use the **print** command.

The following example prints the "srcip" and "dstip" fields from a stream of packets stored in a tcpdump-format file named "/tmp/dump":

```
smacqq 'print srcip, dstip from pcapfile("/tmp/dump")'
```

Nested queries are also supported. For example:

```
print srcip, dstip from (uniq srcip, dstip from pcap-
file("/tmp/dump"))
```

"Where" clauses are supported for both boolean tests based on <, <=, >, >=, =, !=, or arbitrary filtering functions.

```
print dstip from pcapfile("/tmp/dump") where srcip = 128.129.1.2

print dstip from pcapfile("/tmp/dump") where mask(srcip,
128.129.0.0/16)
```

Aliasing with "as" is supported:

```
print dstip, totalbytes from select dstip, sum(len) as totalbytes
from pcapfile("/tmp/dump")
```

Extended relational algebra provides aggregate functions and the "group by" and "having" terms. This syntax is supported, but with slightly different semantics. For example, the following query would behave as it would in SQL:

```
print dstip, sum(len) from pcapfile("/tmp/dump") group by dstip
```

but is also the same as:

```
print dstip, sum from (groupby dstip, '--', sum, len from pcap-
file("/tmp/dump"))
```

However the use of a function (such as "sum()") as one of the arguments will result in the "sum" module processing the data before the "print" module. A module called in this way is expected to annotate the data object with a field of the same name as the function. Thus, the sum field will be part of the object from now on. Thus, it can also be used in subsequent arguments. In addition, however, the module can cause other side-effects to the data. Finally, functions can be used whether or not "group by" is used.

When queries are nested deeply, the syntax described above can become complex. As a result, commas are

optional between arguments, and SMACQ also supports the `|` symbol used in Unix shells. The result is a syntax much more familiar to Unix shell users. Thus the following queries are all equivalent:

```
print srcip, dstip from (uniq srcip, dstip from pcap-  
file("/tmp/dump"))
```

```
print srcip, dstip from uniq srcip, dstip from pcapfile  
"/tmp/dump"
```

```
print srcip dstip from uniq srcip dstip from pcapfile "/tmp/dump"
```

```
pcapfile "/tmp/dump" | uniq srcip dstip | print srcip dstip
```

.

SMACQ is an extensible system that the user can add modules to. See the **smacqp(1)** manpage for a detailed description of modules. Many modules take flags which, like all arguments, can be separated with commas or spaces:

```
print -v, srcip, dstip from pcapfile("/tmp/dump")
```

BUGS

Joins are not yet supported. They will be supported in multiple forms: joins within a buffer window, joins between nondecreasing streams, etc.