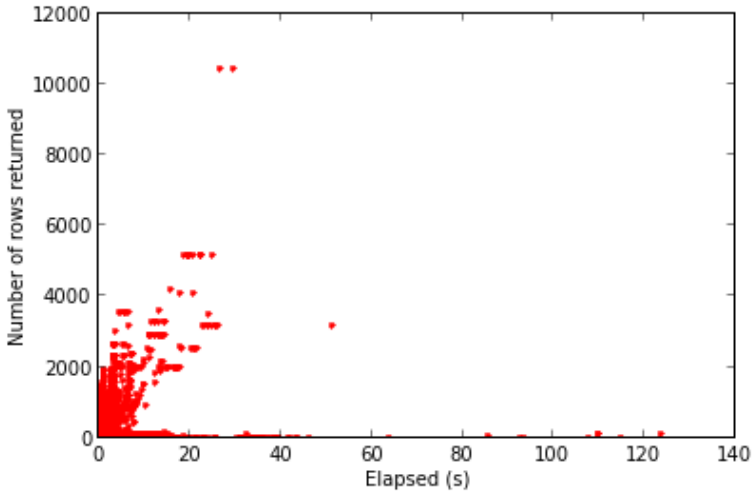
In [1]:
```
import pandas as pd
store = pd.HDFStore('/Volumes/FreshBooks/data/store.h5')
may07 = store['may07']
may08 = store['may08']
```

Is there a correlation between response time and response size?

We can see how queries that return more data take longer to run using the pylab `plot()` method:

In [2]:
```
xlabel('Elapsed (s)')
ylabel('Number of rows returned')
plot(may07['elapsed'], may07['rows'], 'r.', label='May 7')
```

Out[2]:
```
[<matplotlib.lines.Line2D at 0x111ac7410>]
```



But that's a little generic, as it's looking at *all* requests. We can do better! And we can compare one day to another!

Let's start by selecting only a subset of the data. We do deployments around 9am, so we'll just look at requests that come in after release:

In [3]:
```
from datetime import datetime
import pytz
toronto = pytz.timezone('America/Toronto')
may07_0900 = toronto.localize(
    datetime(2012, 5, 7, 9, 0)
).astimezone(pytz.utc).replace(tzinfo=None)
may08_0900 = toronto.localize(
    datetime(2012, 5, 8, 9, 0)
).astimezone(pytz.utc).replace(tzinfo=None)
```

I deployed some interesting changes on the 8th, with the goal of improving performance for invoice-related requests for larger amounts of data. (With thanks to Mike Pirnat for his presentation on metaclassing two years ago!) Let's start by getting the comparison data from the previous day:

In [4]:
```
# Select only the requests that came in after 9am local time
may07_after_0900 = may07[may07.index >= may07_0900]
# Select only the requests for invoice things
may07_invoice_after_0900 = may07_after_0900[
    may07_after_0900['silo'] == 'invoice_InvoiceResource']
# Note that Pandas stores the DatetimeIndex values
# as naïve datetimes in UTC!
may07_invoice_after_0900
```

Out[4]:
```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 284115 entries, 2012-05-07 13:00:00 to 2012-05-08 03:59:59
Data columns:
request_id    284115  non-null values
port          284115  non-null values
accountid     284115  non-null values
userid        284115  non-null values
contactid     284115  non-null values
level         284115  non-null values
silo          284115  non-null values
method        284115  non-null values
rows          284115  non-null values
queries       284115  non-null values
query_time    284115  non-null values
elapsed       284115  non-null values
user          284115  non-null values
sys           284115  non-null values
dtypes: float64(2), int64(10), object(2)
```

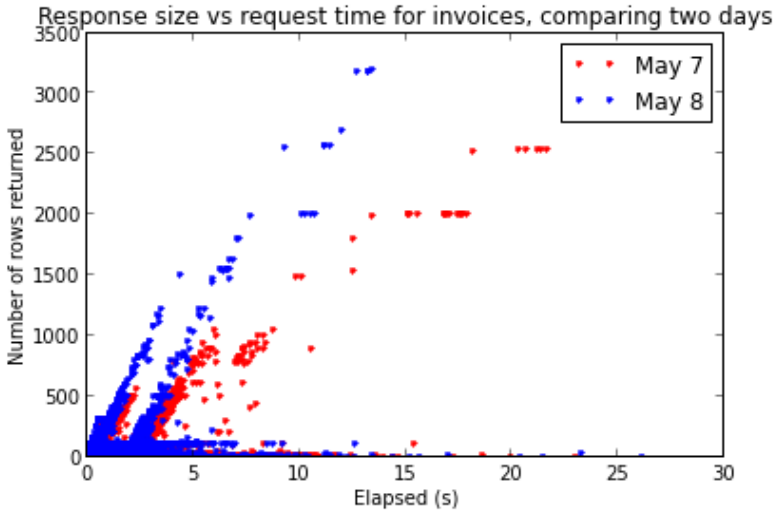
Now we'll get the same range of data, but from after my code went live:

In [5]:
```
may08_after_0900 = may08[may08.index >= may08_0900]
may08_invoice_after_0900 = may08_after_0900[
    may08_after_0900['silo'] == 'invoice_InvoiceResource']
may08_invoice_after_0900
```

Out[5]:
```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 263552 entries, 2012-05-08 13:00:00 to 2012-05-09 03:59:59
Data columns:
request_id    263552  non-null values
port          263552  non-null values
accountid     263552  non-null values
userid        263552  non-null values
contactid     263552  non-null values
level         263552  non-null values
silo          263552  non-null values
method        263552  non-null values
rows          263552  non-null values
queries       263552  non-null values
query_time    263552  non-null values
elapsed       263552  non-null values
user          263552  non-null values
sys           263552  non-null values
dtypes: float64(2), int64(10), object(2)
```

In [6]:
```
title(
    'Response size vs request time for invoices, '
    'comparing two days')
xlabel('Elapsed (s)')
xlim(0, 30)  # There are some uninteresting outliners
ylabel('Number of rows returned')
plot(
    may07_invoice_after_0900['elapsed'],
    may07_invoice_after_0900['rows'], 'r.', label='May 7')
plot(
    may08_invoice_after_0900['elapsed'],
    may08_invoice_after_0900['rows'], 'b.', label='May 8')
legend()
```

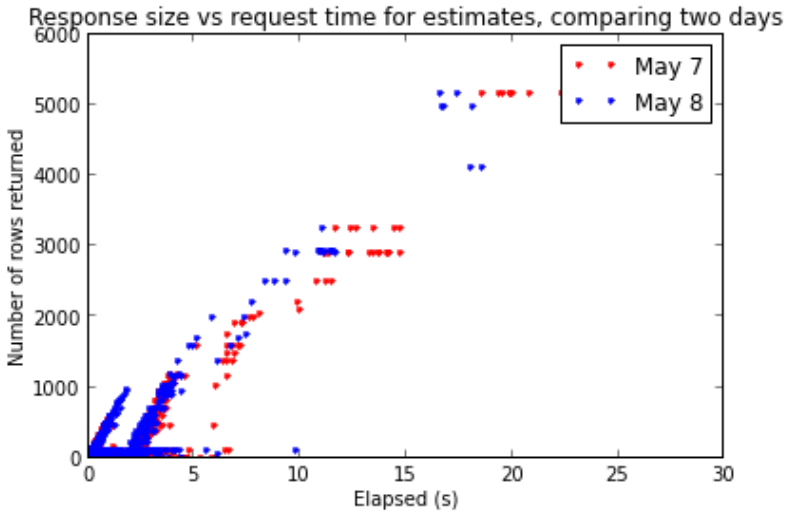
Out[6]: <matplotlib.legend.Legend at 0x11198cf10>



For comparison, I didn't change much about Estimates:

In [7]:
```
may07_estimate_after_0900 = may07_after_0900[
    may07_after_0900['silo'] == 'estimate_EstimateResource']
may08_estimate_after_0900 = may08_after_0900[
    may08_after_0900['silo'] == 'estimate_EstimateResource']
title(
    'Response size vs request time for estimates, '
    'comparing two days')
xlabel('Elapsed (s)')
xlim(0, 30)
ylabel('Number of rows returned')
plot(
    may07_estimate_after_0900['elapsed'],
    may07_estimate_after_0900['rows'], 'r.', label='May 7')
plot(
    may08_estimate_after_0900['elapsed'],
    may08_estimate_after_0900['rows'], 'b.', label='May 8')
legend()
```

Out[7]: <matplotlib.legend.Legend at 0x139780f90>



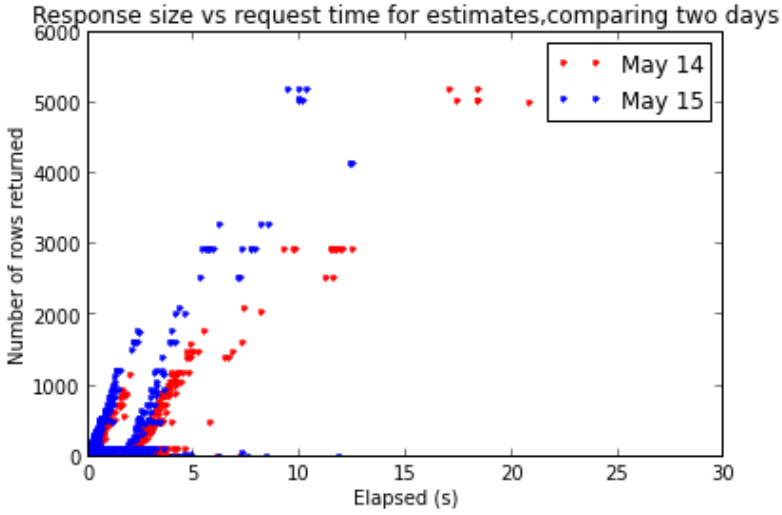
But I did change estimates week later! Let's see how they fare:

In [8]:
```
may14 = store['may14']
may14_0900 = toronto.localize(
    datetime(2012, 5, 14, 9, 0)
).astimezone(pytz.utc).replace(tzinfo=None)
may14_after_0900 = may14[may14.index >= may14_0900]
may14_estimate_after_0900 = may14_after_0900[
    may14_after_0900['silo'] == 'estimate_EstimateResource']

may15 = store['may15']
may15_0900 = toronto.localize(
    datetime(2012, 5, 15, 9, 0)
).astimezone(pytz.utc).replace(tzinfo=None)
may15_after_0900 = may15[may15.index >= may15_0900]
may15_estimate_after_0900 = may15_after_0900[
    may15_after_0900['silo'] == 'estimate_EstimateResource']

title(
    'Response size vs request time for estimates,'
    'comparing two days')
xlabel('Elapsed (s)')
xlim(0, 30)
ylabel('Number of rows returned')
plot(
    may14_estimate_after_0900['elapsed'],
    may14_estimate_after_0900['rows'], 'r.', label='May 14')
plot(
    may15_estimate_after_0900['elapsed'],
    may15_estimate_after_0900['rows'], 'b.', label='May 15')
legend()
```

Out[8]: <matplotlib.legend.Legend at 0x1397cd9d0>



Bingo!