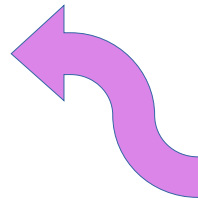
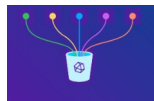


# Data flow with Time Series DB

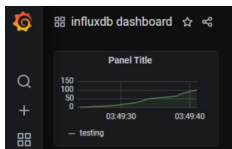
```
curl http://borg:8086/data.cgi  
-F data=@data.txt
```



```
$influx->write(  
  $point, database => $data);
```



```
http://influx:8086/write?db=data  
-d 'swap,host=host 0.85  
1656472338'
```



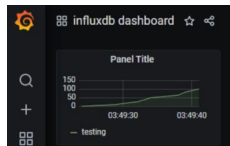
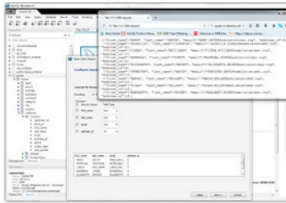
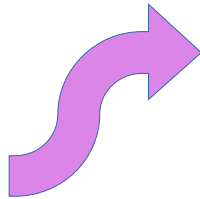
influxdb Source  
<http://influx:8086>

# Data flow with classic DB

```
curl http://borg:8086/data.cgi  
-F data=@data.txt
```

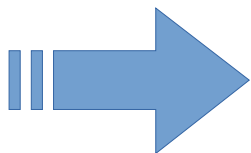
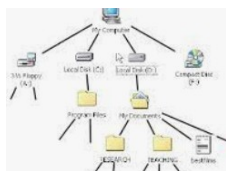


```
$DBI->prepare($query);  
$DBI->execute
```



influxdb Source  
<http://influx:8086>

# Ephemeral Data Point flow with Prometheus



```
go_memstats_heap_alloc_bytes 9.03912e+07
# HELP go_memstats_heap_idle_bytes Number of
# TYPE go_memstats_heap_idle_bytes gauge
go_memstats_heap_idle_bytes 4.0697856e+07
# HELP go_memstats_heap_inuse_bytes Number
# TYPE go_memstats_heap_inuse_bytes gauge
go_memstats_heap_inuse_bytes 9.2798976e+07
# HELP go_memstats_heap_objects Number of
# TYPE go_memstats_heap_objects gauge
```

<https://8443//pwp/prometheus/metrics>



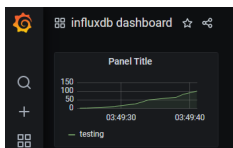
Prometheus

```
scrape_configs:
- job_name: 'application'

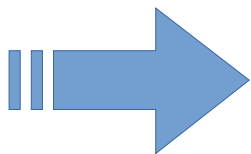
  scrape_interval: 10s
  # metrics_path: /application/metrics
  # reload after change
  metrics_path: /metrics
  honor_labels: true

static_configs:
- targets: ['application:8080']
  labels:
    group: 'application'
```

prometheus Source  
<http://prometheus:9090>



# Data Backfill into InfluxDB or Regular DB



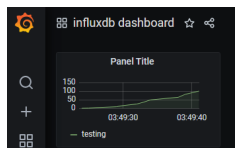
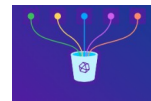
```
connection.prepareStatement(  
    "insert into data values  
    (?, ?, ?)".addBatch()
```



```
influxDB.write(  
    BatchPoints.builder().  
    points().build());
```



prometheus Source  
<http://prometheus:9090>



# Data flow with Influx DB *and* Prometheus (exotic scenario – not to be used)



```
curl http://borg:8086/data.cgi  
-F data=@data.txt
```

```
$influx->write(  
  $point,  
  database => $data  
);
```

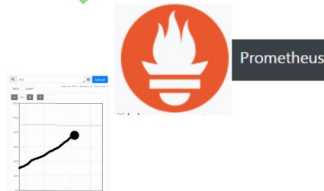
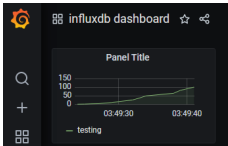


```
http://influx:8086/write?db=data  
-d 'swap,host=host 0.85  
1656472338'
```



*influxdb\_exporter*

prometheus Source  
<http://prometheus:9090>



# Historic Data Ingestion Scenarios

Snapshot scraping style

Green	Light Blue	Pink
Green	Light Blue	Pink
Green	Light Blue	Pink
Brown	Brown	Purple
Light Green	Grey	Light Brown


  

Green	Light Blue	Pink
Green	Light Blue	Pink
Green	Light Blue	Pink
Green	Light Blue	Pink
Brown	Brown	Purple



Timeseries backfill style

Dark Green	Light Blue	Pink
Dark Green	Light Blue	Pink
Dark Green	Light Blue	White
Dark Green	Light Blue	Pink
White	White	Pink

Green	Dark Blue	Pink
Green	Dark Blue	Pink
Green	Dark Blue	White
Green	Dark Blue	Pink
White	White	Pink

# InfluxDB 1.x vs. 2.x

## SQL Query style

```
db = InfluxDBFactory.connect host user, pass
point = Point.measurement seriesName
    .time System.currentTimeMillis()
    .tag "hostname", hostname
    .addField "idle", 40L
    .build()
db.writePoint point
BatchPoints.builder().points(points).build
db.write batchpoints
db.write "cpu,hostname=host value=60.0"
```

```
influxDB = InfluxDBFactory.connect
query = new Query queryString, databaseName
response = influxDB.query "select * from
testing"
results = response.getResults
while results.iterator.hasNext
```

## NoSQL Style

```
client = InfluxDBClientFactory.create host, token, org
api = client.getWriteApiBlocking
api.writePoint
    Point.measurement "cpu"
        .addTag "hostname", "hostname"
        .addField "value", 55D
        .time Instant.now()

api.writeRecord
    "cpu,hostname=host value=60.0"

api.writeMeasurement new POJO
```

```
client = InfluxDBClientFactory.create host, token, org
api = client.getInfluxQLQueryApi
tables = queryApi.query
    from(bucket:testbucket)
        |> range(start: 0)
        |> filter(fn: (r) => r["_measurement"] == "cpu")
records = table.getRecords
record.getValueByKey
```