Data Pipeline with Threading

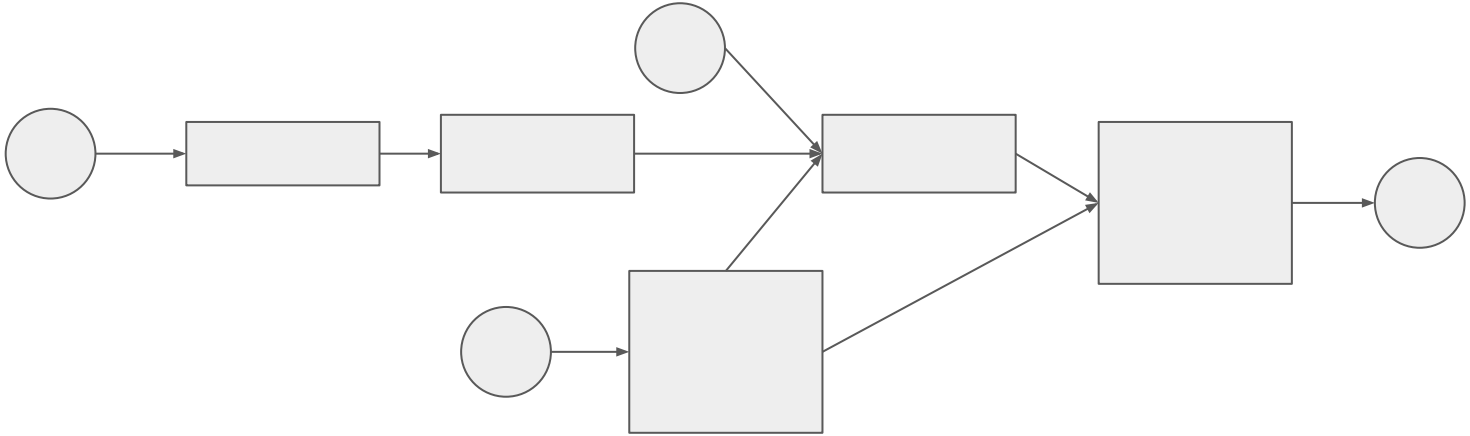
Ken Q Pu, Faculty of Science, Ontario Tech University

Comprehensive Guide

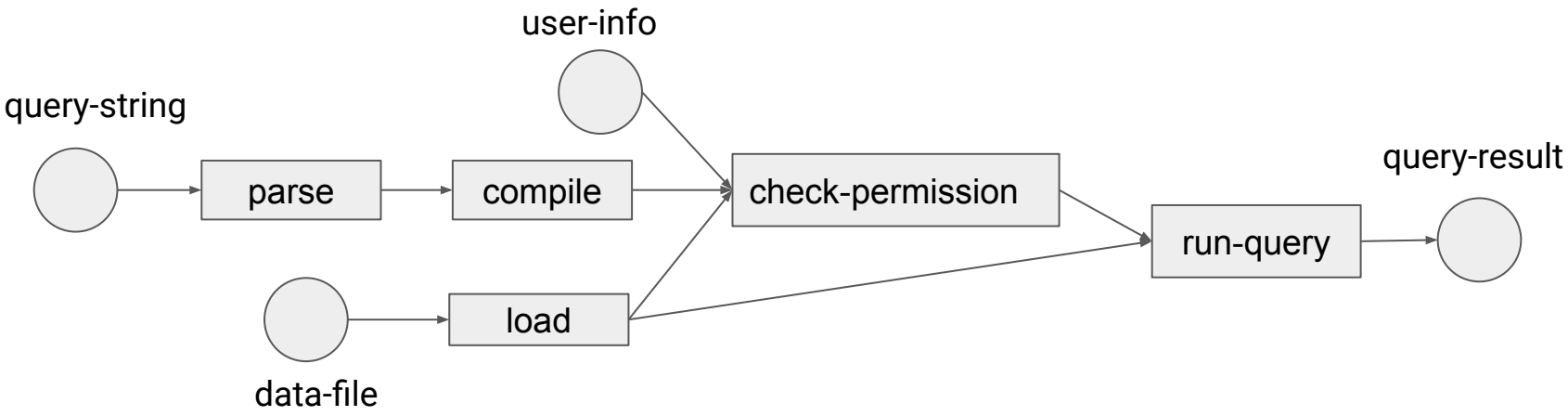
https://clojure.org/api/cheatsheet

Clojure's perspective of computation

Functional programming languages including Clojure focus on data generation and transformation.



Clojure's perspective of computation



Clojure's perspective of computation

```
(run-query
 (check-permission
  (compile (parse query-string))
  (load data-file)
  user-info)
 (load data-file))
```

Clojure provides syntactic support to help with building complex data pipelines.

Thread-first macro form

```
(-> <initial-value> <form> <form> ... <form>)
```

result

The previous result is inserted as the ***first*** *argument of the next form.*

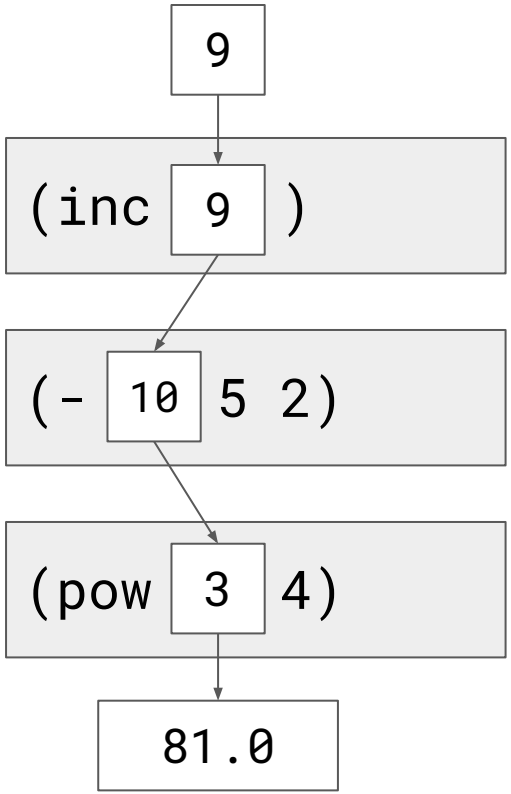
```
(-> 9 (inc) (- 5 2) (pow 4))

=> ?
```

Thread-first macro form

```
(-> 9 (inc) (- 5 2) (pow 4))

=> ?
```



Thread-last macro form

```
(->> <initial-value> <form> <form> ... <form>)
```

result

The previous result is inserted as the ***last*** *argument of the next form.*

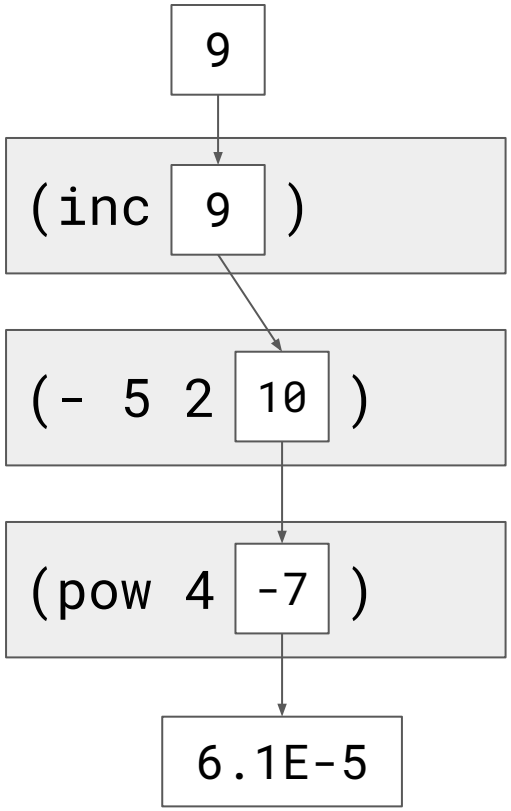
```
(->> 9 (inc) (- 5 2) (pow 4))

=> ?
```

Thread-last macro form

```
(->> 9 (inc) (- 5 2) (pow 4))

=> ?
```



as-> threading form

Thread-first and thread-last fix the position to insert the previous value in the next form throughout the threading form.

The as-> form allows one to control where the insertion occurs throughout the threading form.

```
(as-> <initial-value> <symbol>
      <form>
      <form> ...)
```

The forms use the symbol to indicate where the previous result should be inserted.

as-> macro form

```
(as-> 9 x
 (inc x)
 (- 5 x 2)
 (pow x 4))

=> ?
```

9

```
(inc  x  )
```

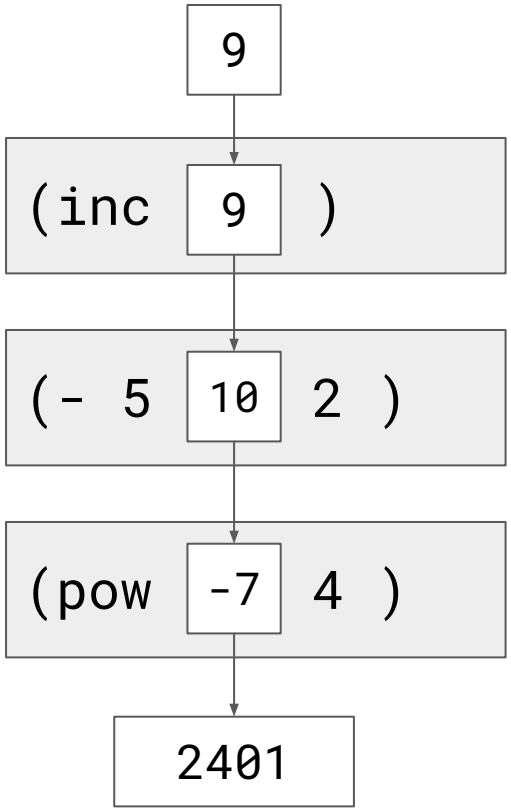
```
(- 5  x  2 )
```

```
(pow  x  4 )
```

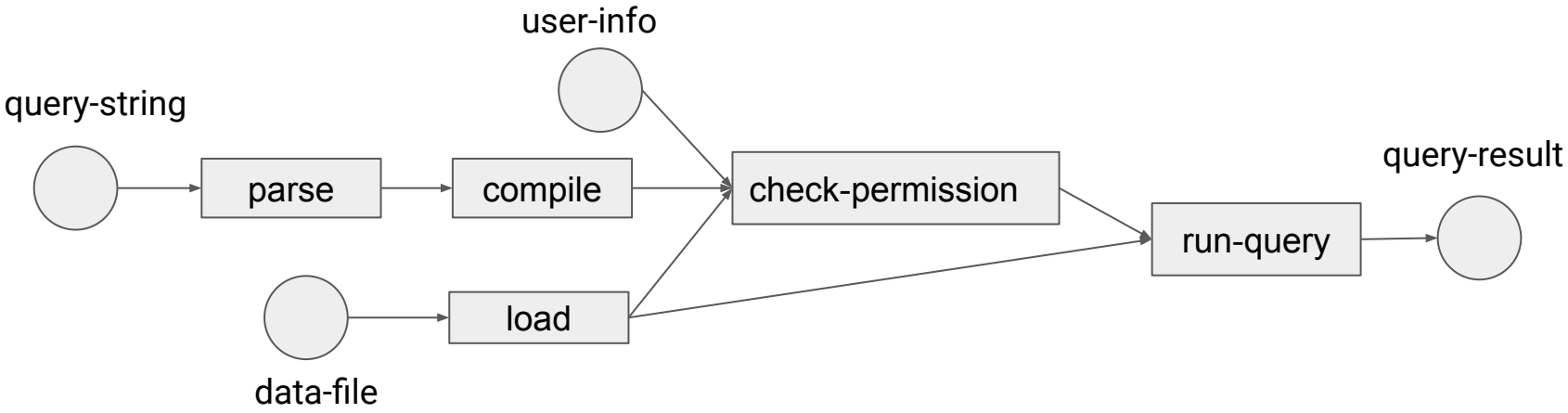
as-> macro form

```
(as-> 9 x
 (inc x)
 (- 5 x 2)
 (pow x 4))

=> ?
```

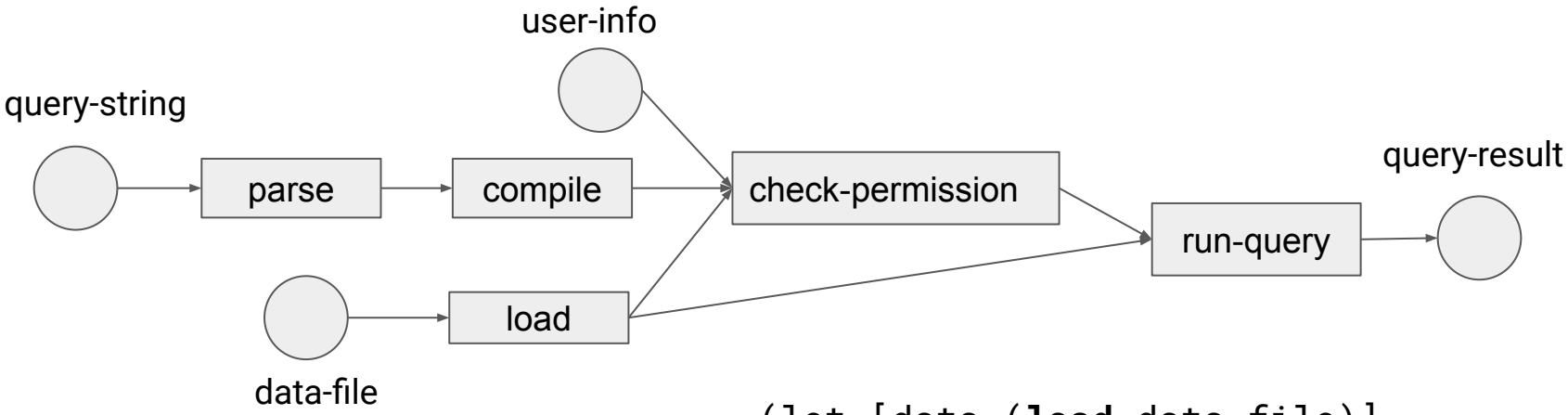


Complex data pipelines




```
(run-query
 (check-permission
  (compile (parse query-string))
  (load data-file)
  user-info)
 (load data-file))
```

Complex data pipelines




```
(let [data (load data-file)]
  (-> query-string
      (parse)
      (compile)
      (check-permission user-info data)
      (run-query data)))
```