Delta Encodings

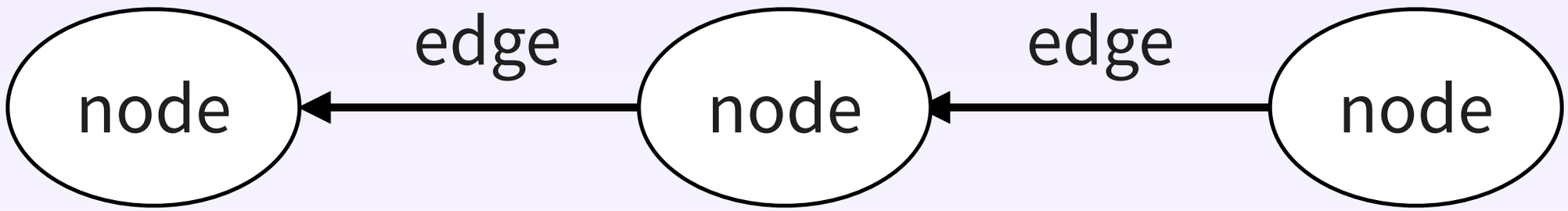
Heinrich Apfelmus

IOHK internal, 15 Sep 2021

Data.Chain

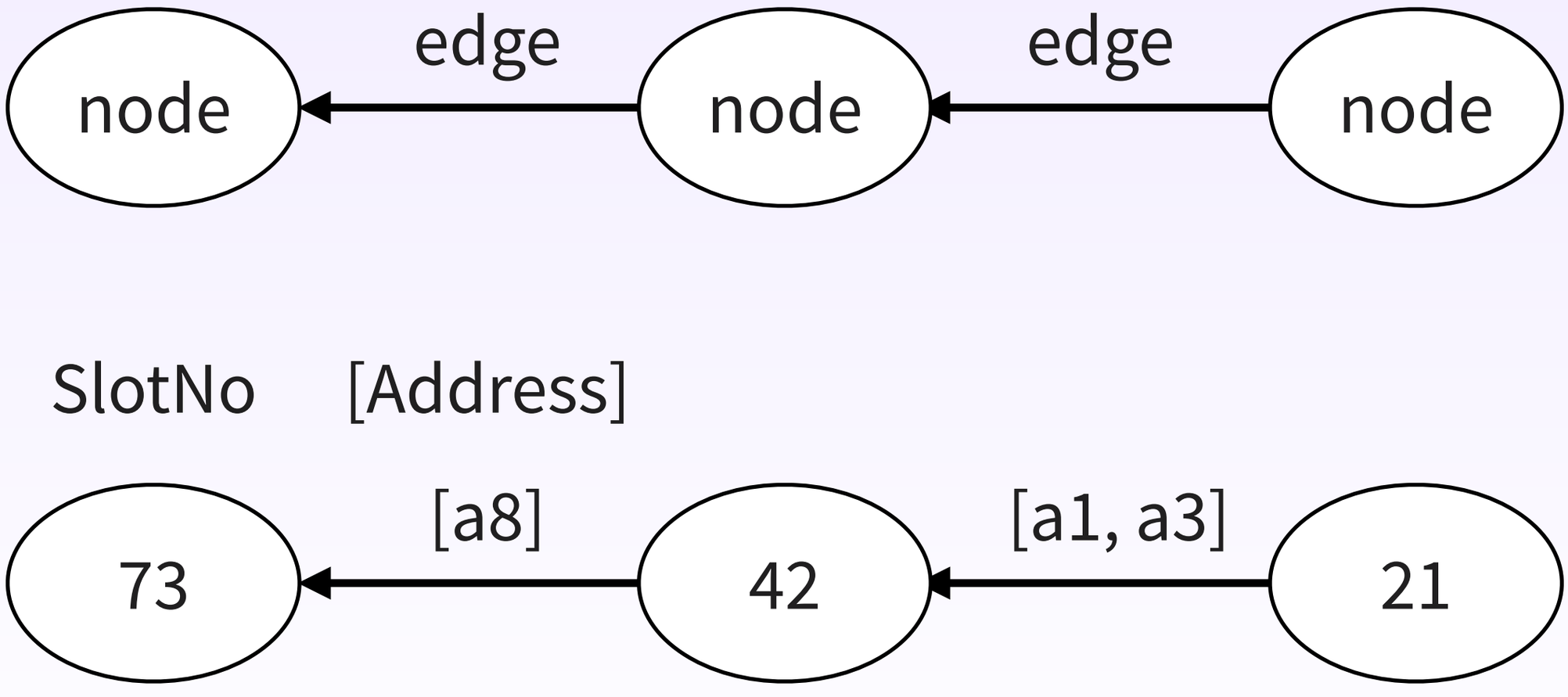
Chain

```
data Chain node edge
```



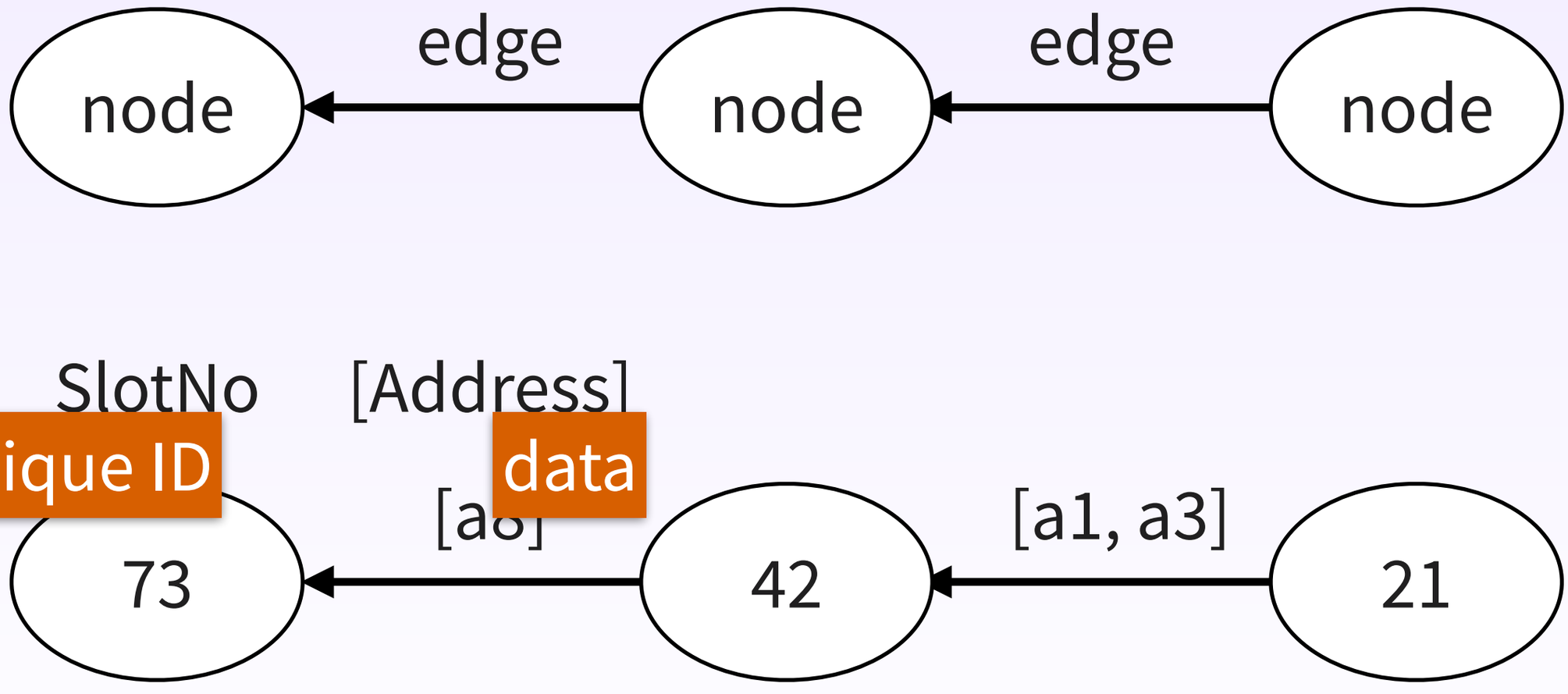
Chain

```
data Chain node edge
```



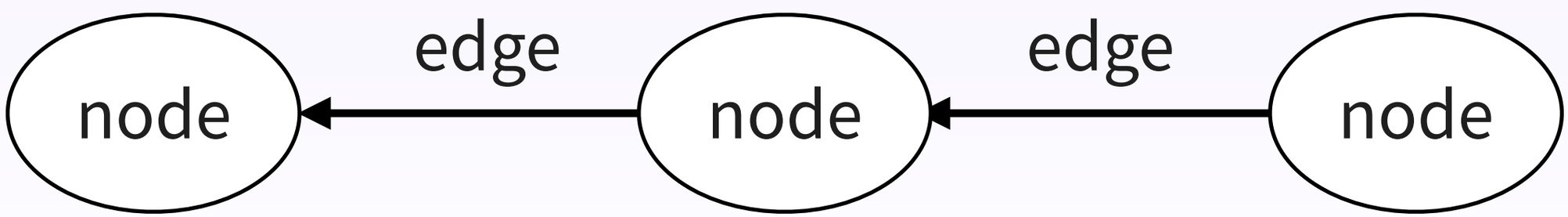
Chain

```
data Chain node edge
```



DeltaChain

```
data DeltaChain node edge
     = AppendTip node edge
     | CollapseNode node
     | RollbackTo node
```



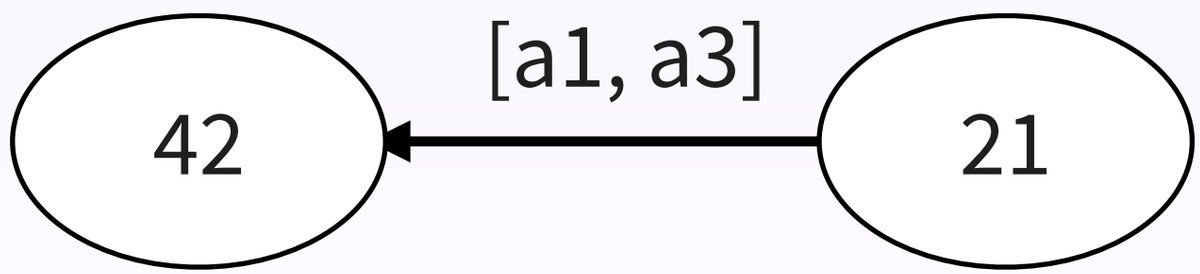
DeltaChain

```
data DeltaChain node edge
     = AppendTip node edge
     | CollapseNode node
     | RollbackTo node
```

DeltaChain

```
data DeltaChain node edge
     = AppendTip node edge
     | CollapseNode node
     | RollbackTo node
```

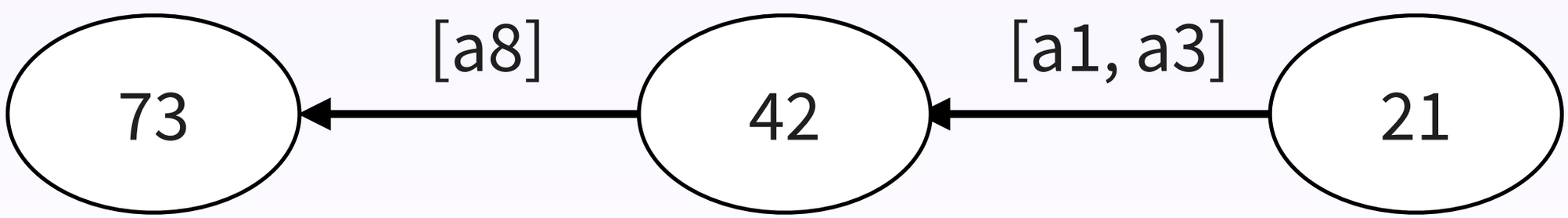
```
AppendTip 73 [a8]
```



DeltaChain

```
data DeltaChain node edge
     = AppendTip node edge
     | CollapseNode node
     | RollbackTo node
```

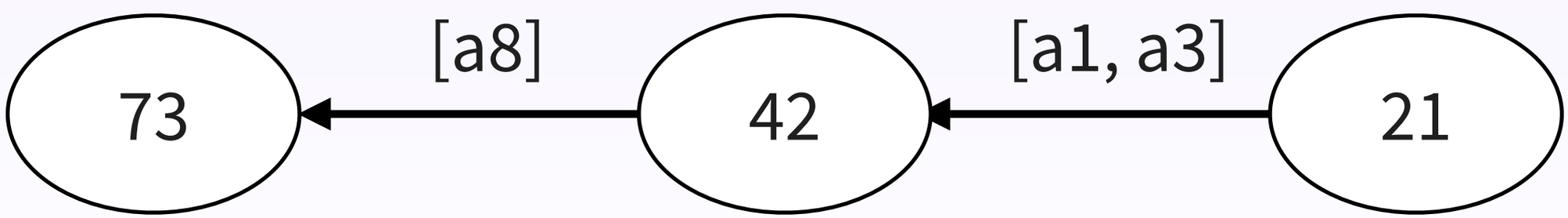
```
AppendTip 73 [a8]
```



DeltaChain

```
data DeltaChain node edge
     = AppendTip node edge
     | CollapseNode node
     | RollbackTo node
```

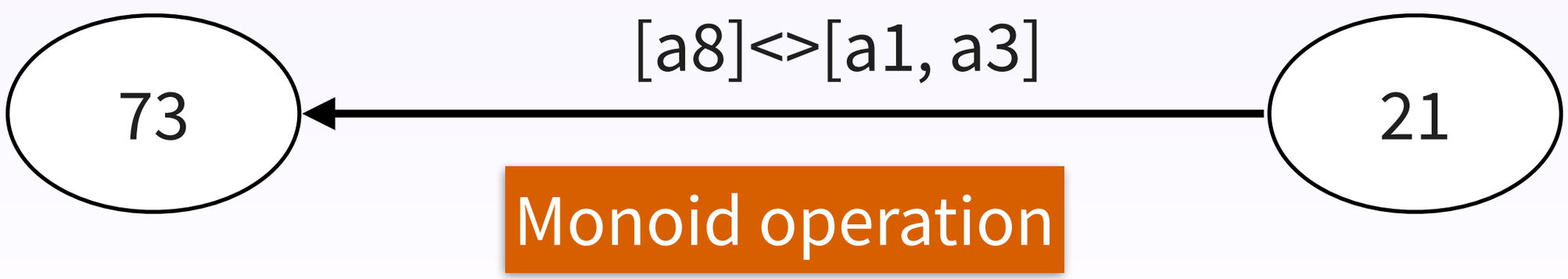
```
CollapseNode 42
```



DeltaChain

```
data DeltaChain node edge
     = AppendTip node edge
     | CollapseNode node
     | RollbackTo node
```

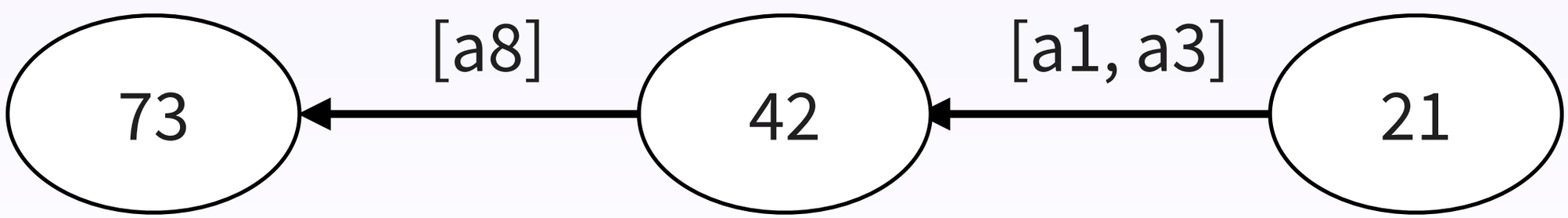
```
CollapseNode 42
```



DeltaChain

```
data DeltaChain node edge
     = AppendTip node edge
     | CollapseNode node
     | RollbackTo node
```

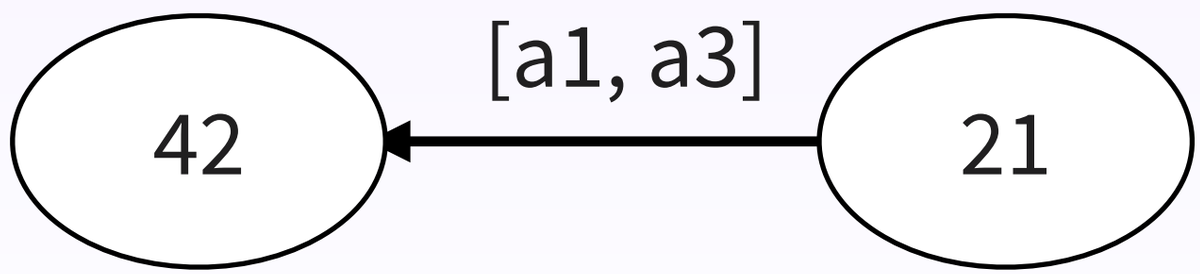
```
RollbackTo 42
```



DeltaChain

```
data DeltaChain node edge
     = AppendTip node edge
     | CollapseNode node
     | RollbackTo node
```

```
RollbackTo 42
```



Table

Table

```
data Table row
```

(Int)	row
1	...
2	...
3	...

Table

```
data Table row
```

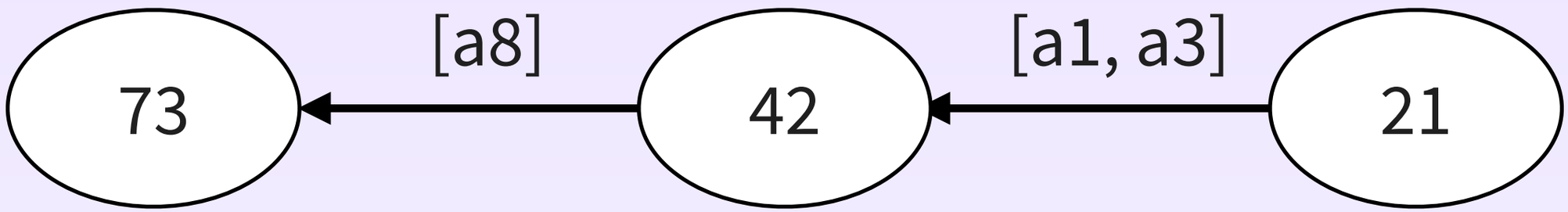
(Int)	row
1	…
2	…
3	…

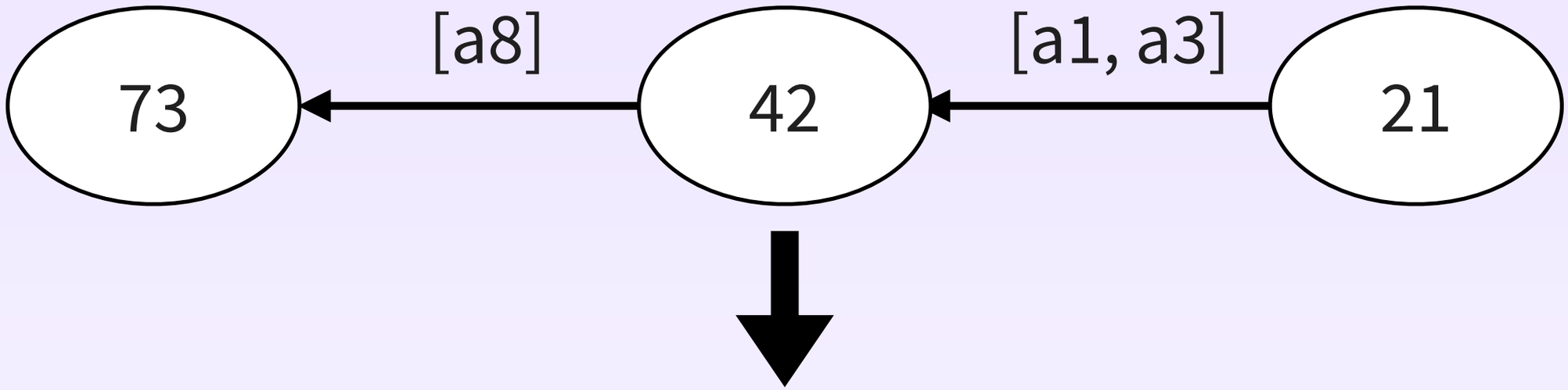
```
empty :: Table row
```

```
insertMany :: [row]
→ Table row → Table row
```

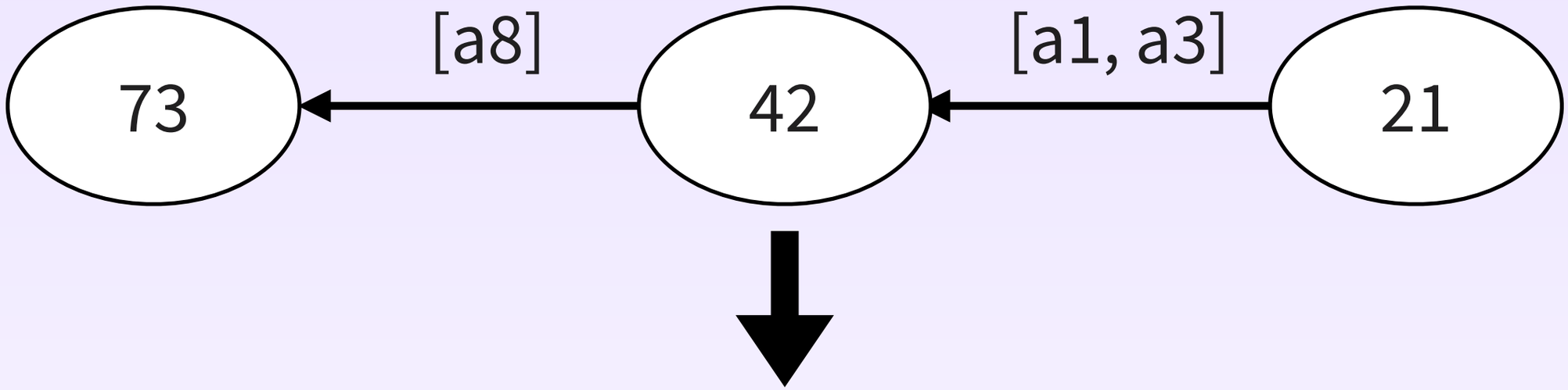
```
deleteWhere
:: (row → Bool)
→ Table row → Table row
```

Embedding
Chain → Table


73
[a8]
42
[a1, a3]
21

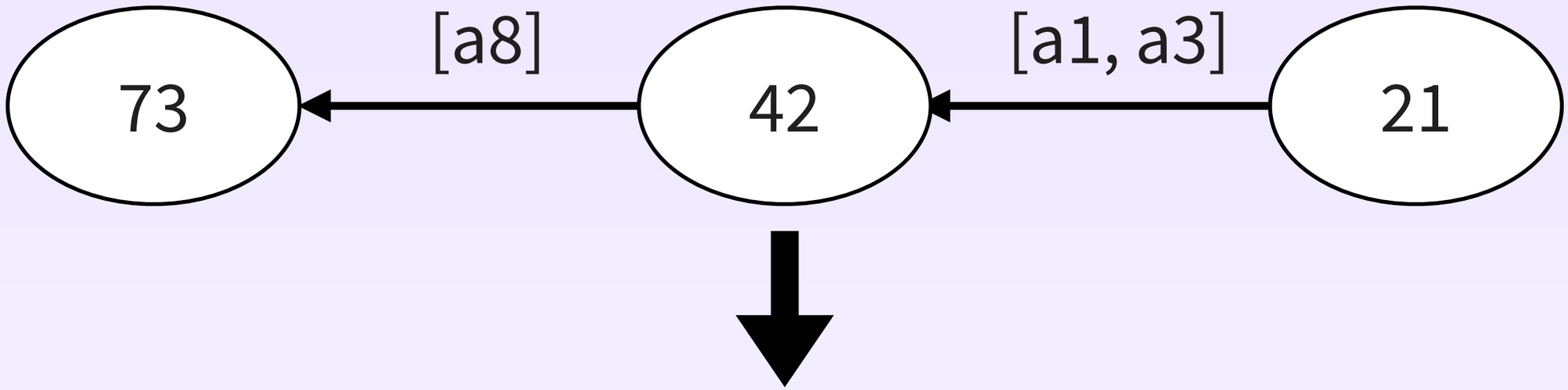


(Int)	to	from	via
1	42	21	a1
2	42	21	a3
3	73	42	a8



(Int)	to	from	via
1	42	21	a1
2	42	21	a3
3	73	42	a8

```
data Edge n e =
  Edge { to :: n, from :: n, via :: e }
```



(Int)	to	from	via
1	42	21	(a1, 1)
2	42	21	(a3, 0)
3	73	42	(a8, 0)

list position

```
data Edge n e =
  Edge { to :: n, from :: n, via :: e }
```

Embedding

```
write  :: Chain … → Table (Edge …)
```

Embedding

```
write  :: Chain … → Table (Edge …)
update :: Chain … → Table (Edge …)
  → DeltaChain … → DeltaTable …
```

Embedding

```
write  :: Chain … → Table (Edge …)
update :: Chain … → Table (Edge …)
  → DeltaChain … → DeltaTable …
load   :: Table … → Maybe (Chain …)
```

Embedding

```
data Embedding’ da db = Embedding’
 { load   :: Base db → Maybe (Base da)
 , write  :: Base da → Base db
 , update :: Base da → Base db → da → db
 }
```

```
instance  Delta da  where …
instance  Delta db  where …
```

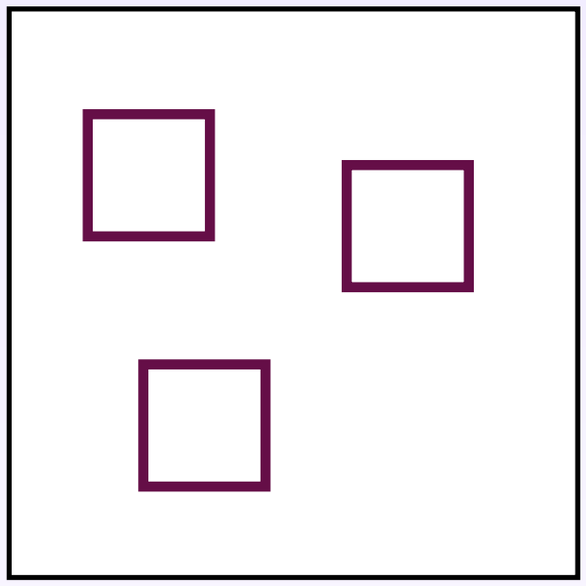
Embedding

```
data Embedding’ da db = Embedding’
 { load   :: Base db → Maybe (Base da)
 , write  :: Base da → Base db
 , update :: Base da → Base db → da → db
 }
```

```
data Embedding da db -- performance
```

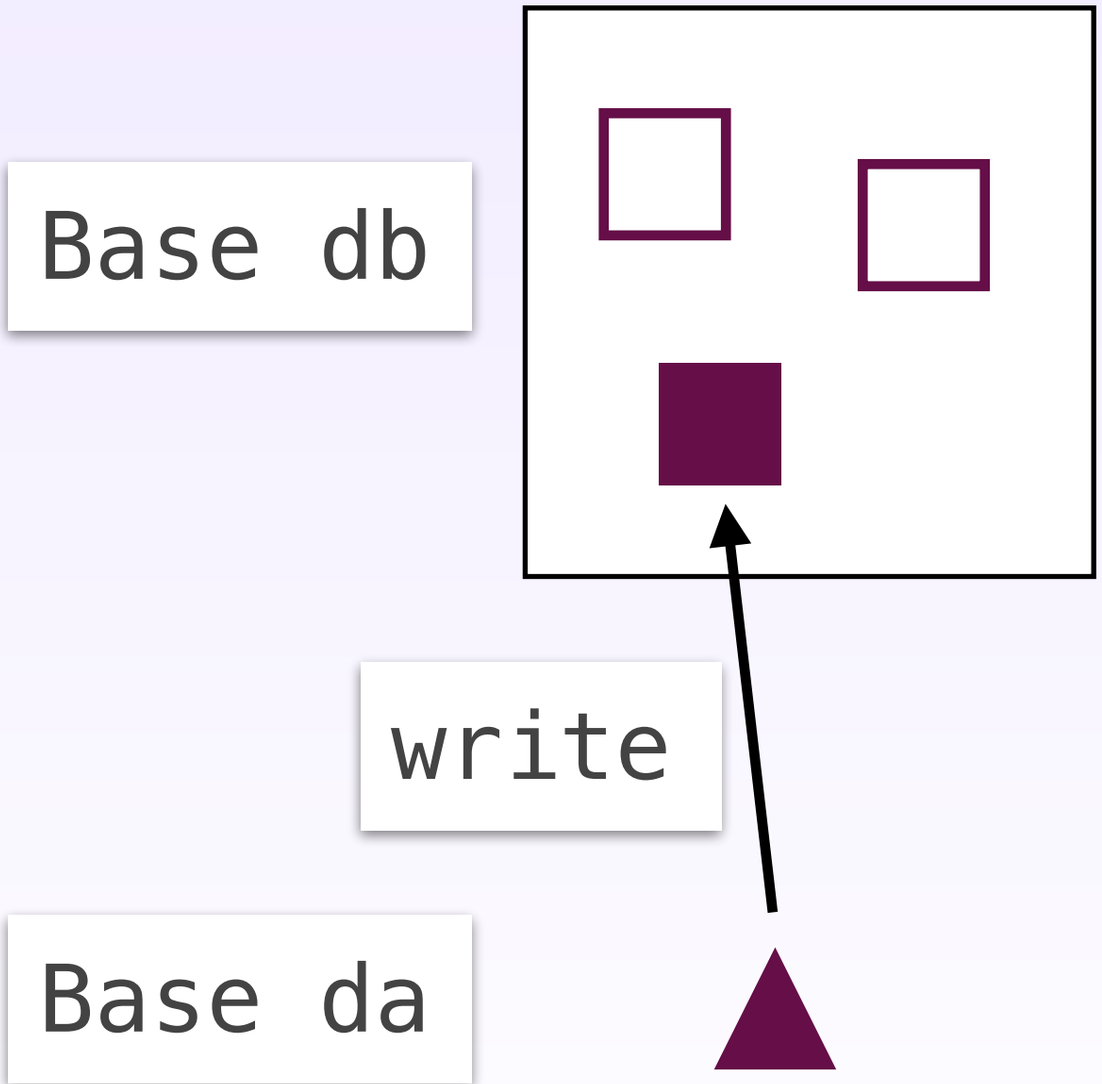
Embedding

Base db

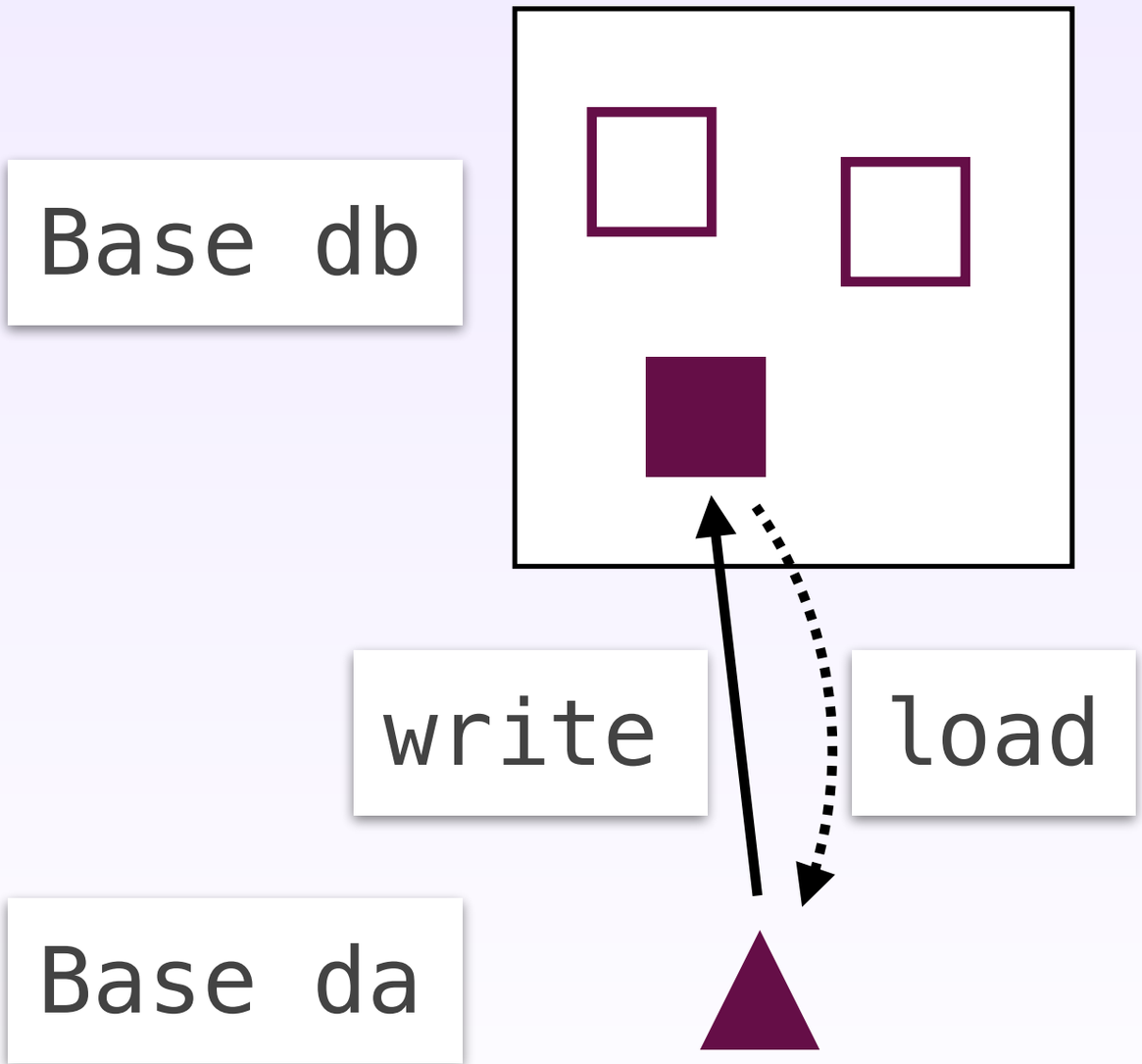


Base da

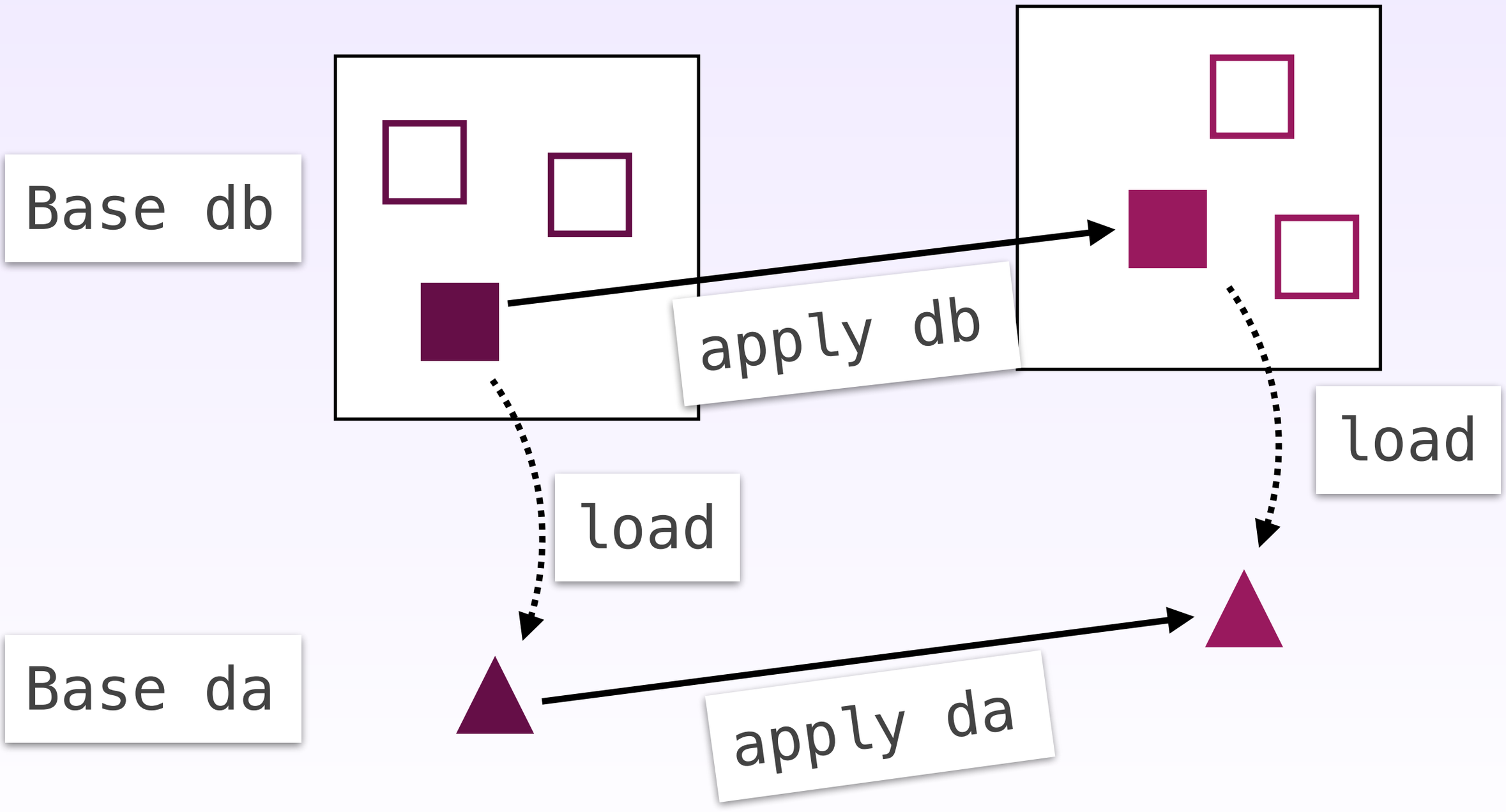
Embedding



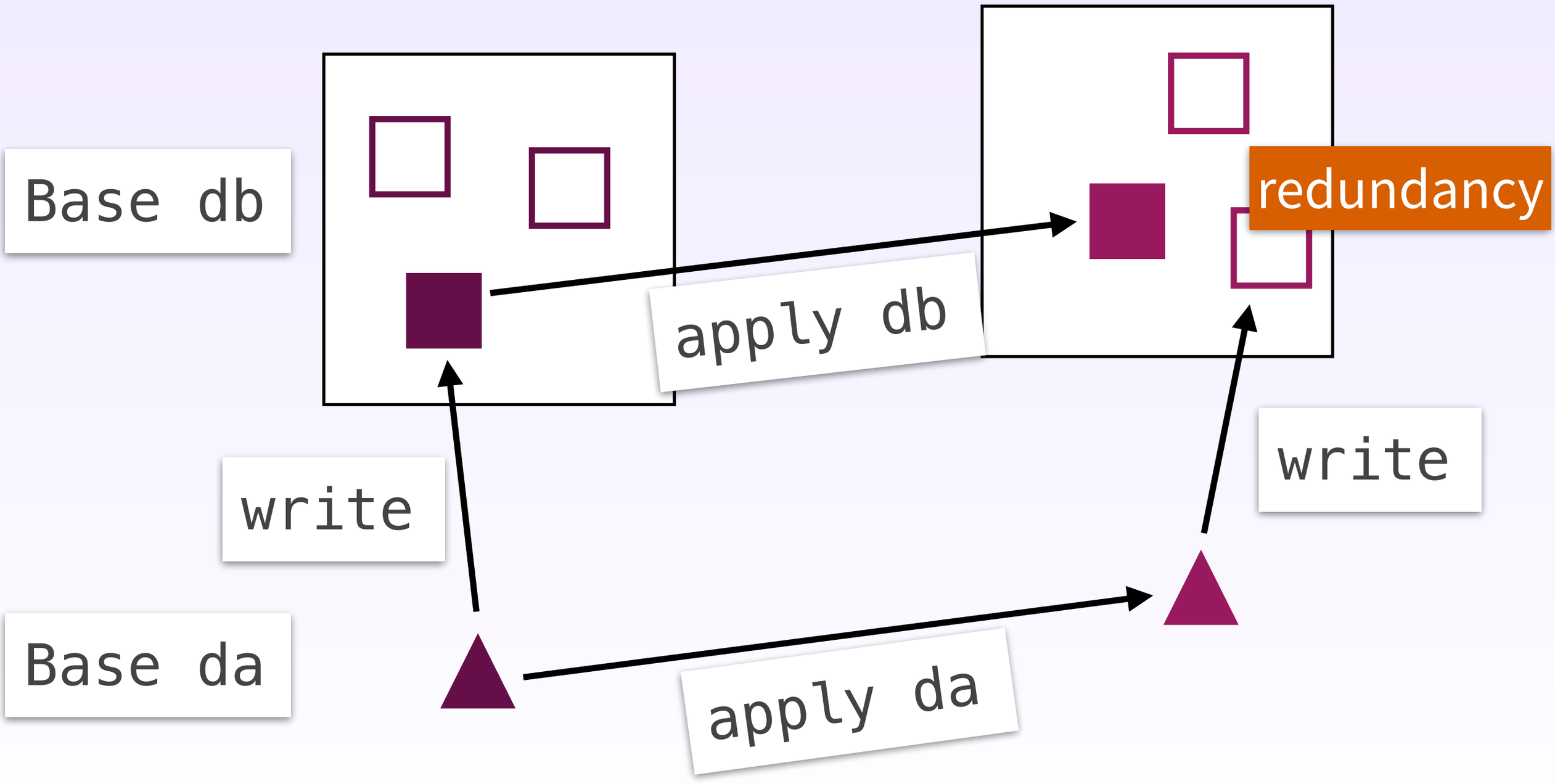
Embedding



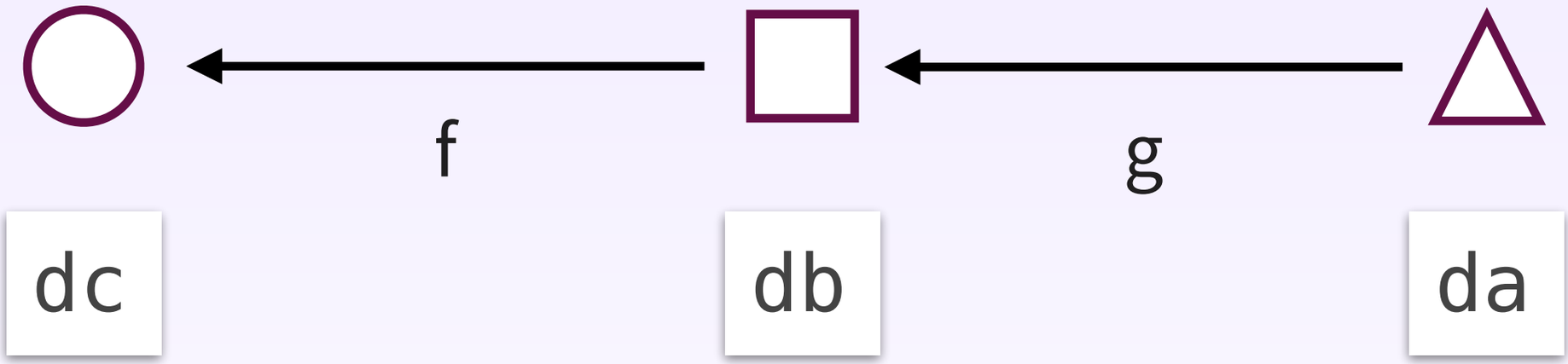
Embedding



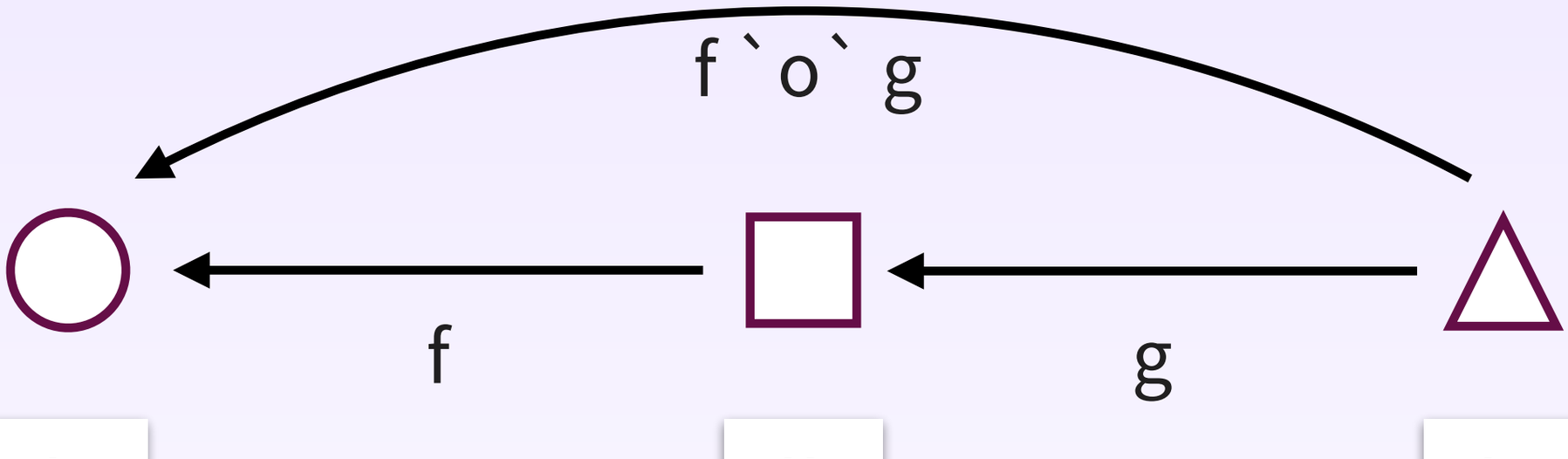
Embedding



Embedding: Composition



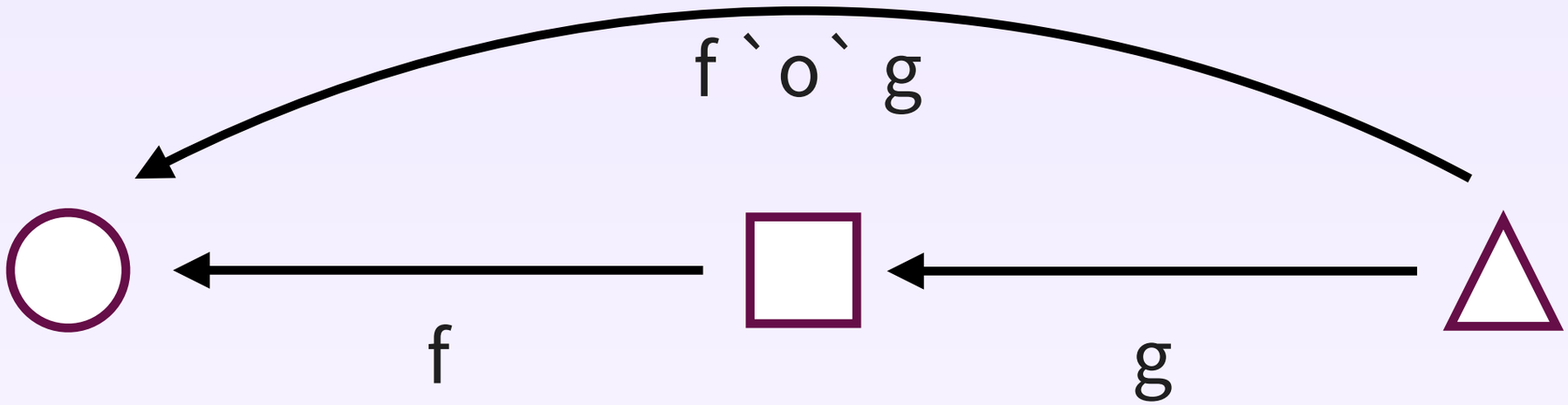
Embedding: Composition



dc db da

```
o :: Embedding db dc → Embedding da db
   → Embedding db dc
```

Embedding: Composition



dc db da

```
o :: Embedding db dc → Embedding da db
   → Embedding db dc
```

```
class Semigroupoid c where  o :: …
```