

TypeDB Fundamentals Lecture Series

TypeDB: the Polymorphic Database



Dr. James Whiteside

Research Engineer, Vaticle

Previously: Computational Solid-State Physicist
@ University of Surrey

Core features of TypeDB

- Conceptual data model
- Type-theoretic language
- Strong type system
- Symbolic reasoning

Conceptual model

Polymorphic schemas

Introducing the PERA model



- Represent independent concepts.
- Can own attributes.
- Can play roles in relations.
- Can be subtyped.
- Can be abstract.

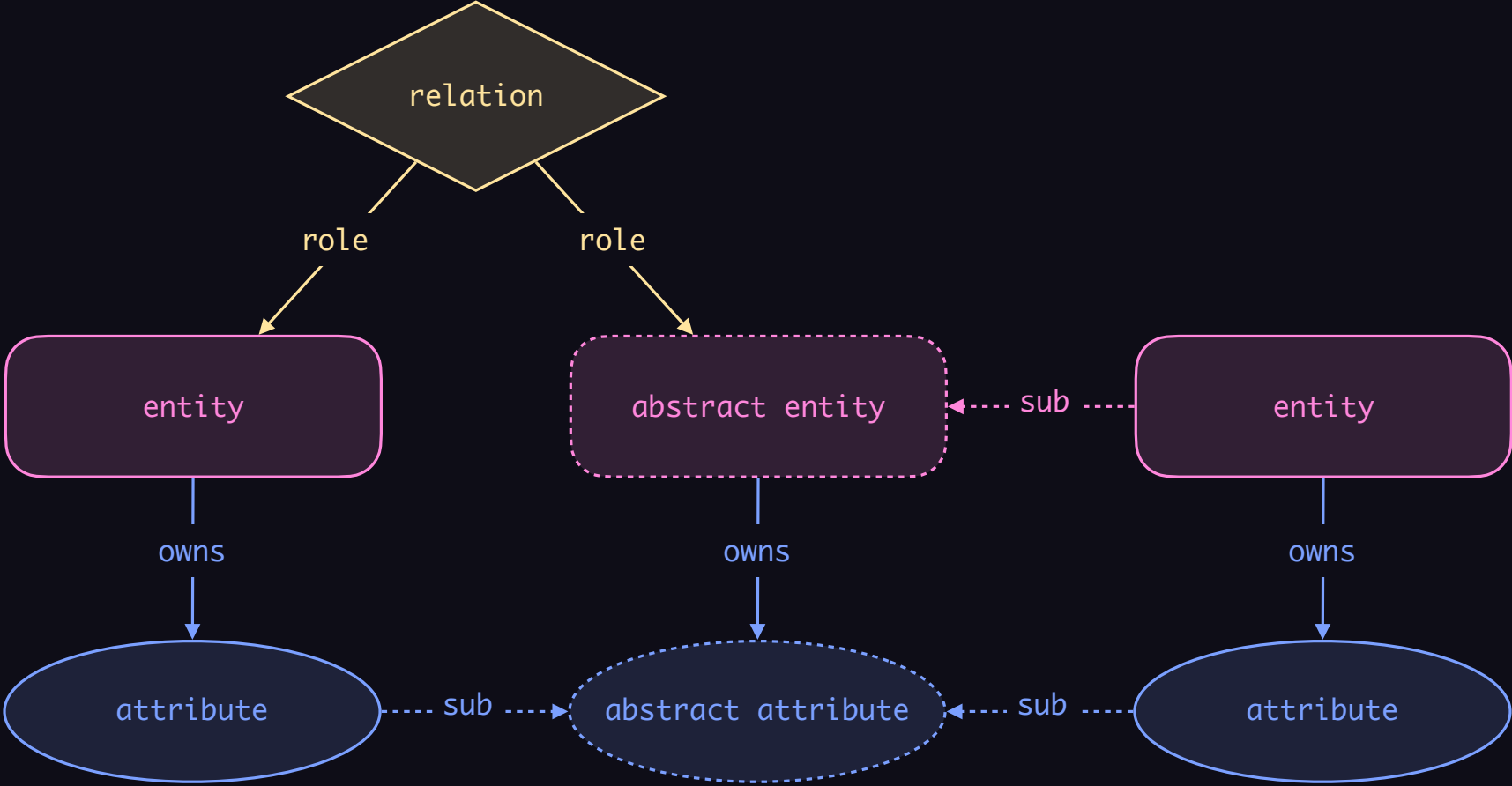


- Represent concepts dependent on other concepts.
- Have one or more roles.
- Can own attributes.
- Can play roles in relations.
- Can be subtyped.
- Can be abstract.

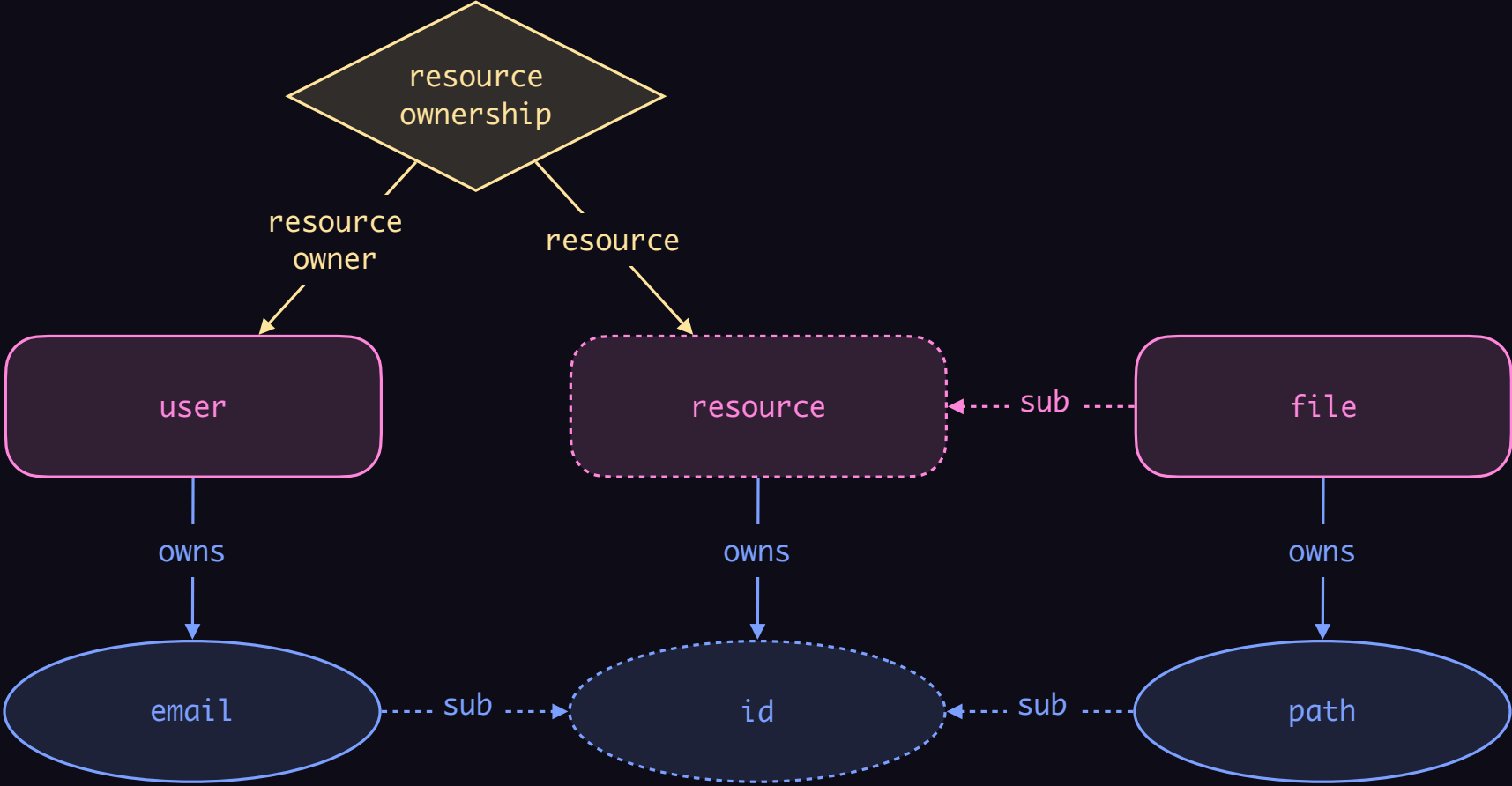


- Represent literal values in a domain.
- Have value types.
- Can be subtyped.
- Can be abstract.

Introducing the P_ER_A model



Introducing the P_ER_A model



Introducing the P ERA model

```
define
```

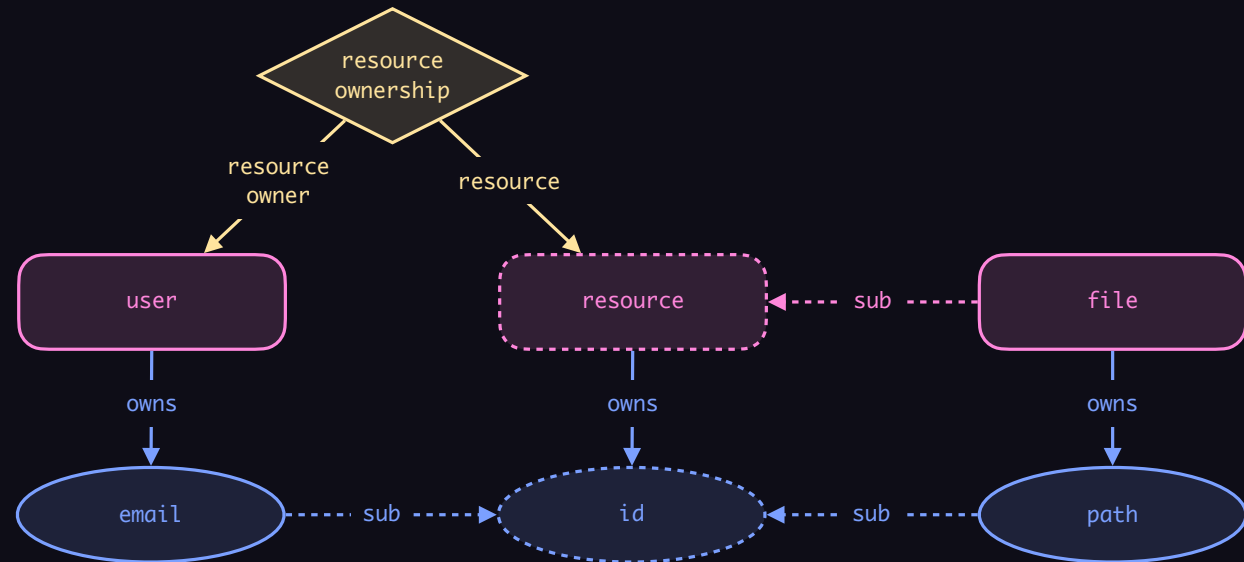
```
user sub entity,  
  owns email,  
  plays resource-ownership:resource-owner;
```

```
resource sub entity, abstract  
  owns id,  
  plays resource-ownership:resource;
```

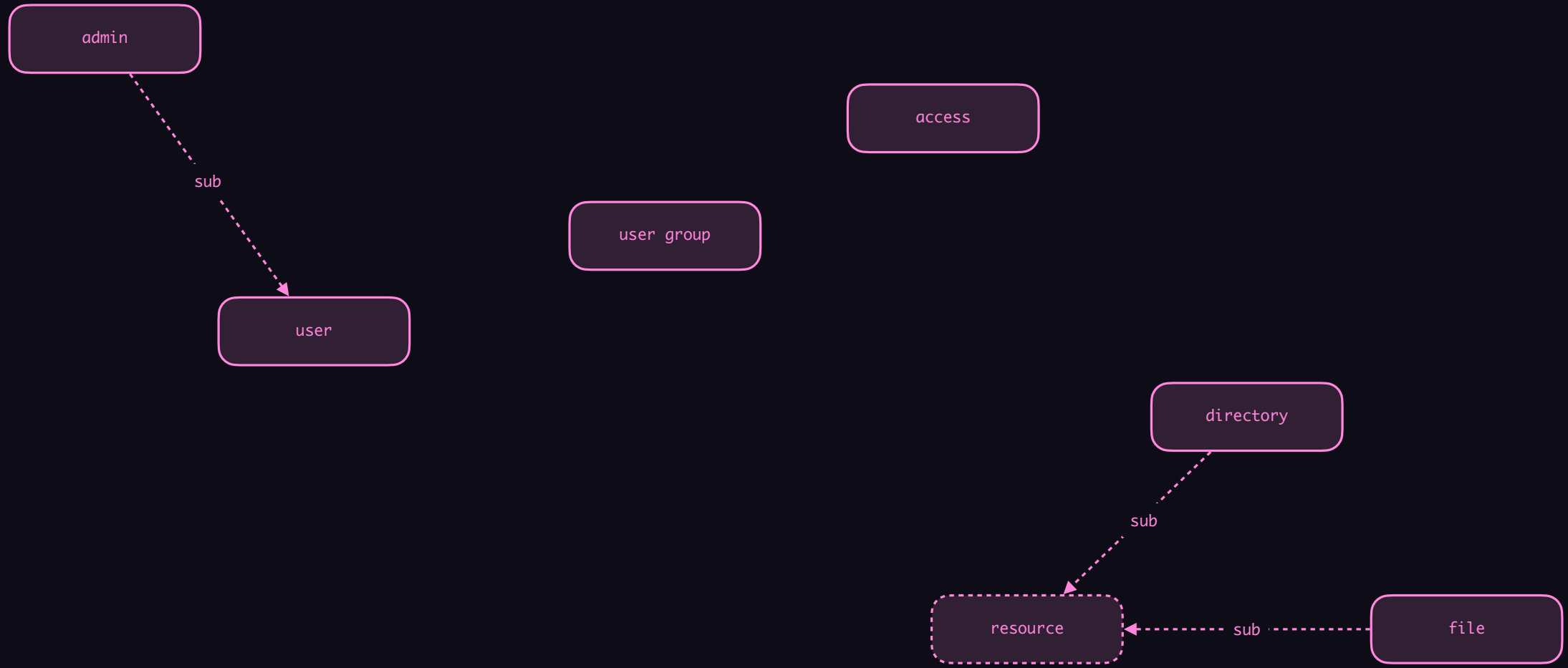
```
file sub resource,  
  owns path as id;
```

```
resource-ownership sub relation,  
  relates resource,  
  relates resource-owner;
```

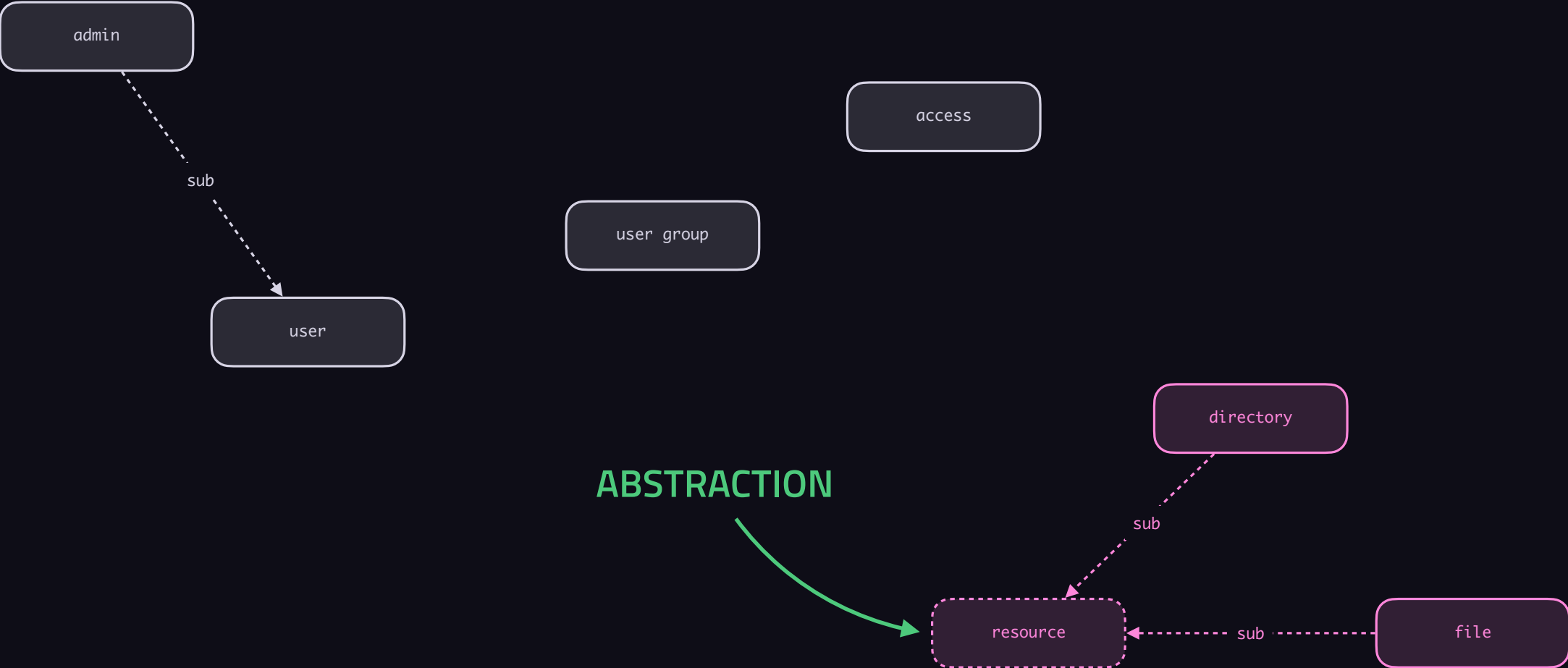
```
id sub attribute, abstract, value string;  
email sub id;  
path sub id;
```



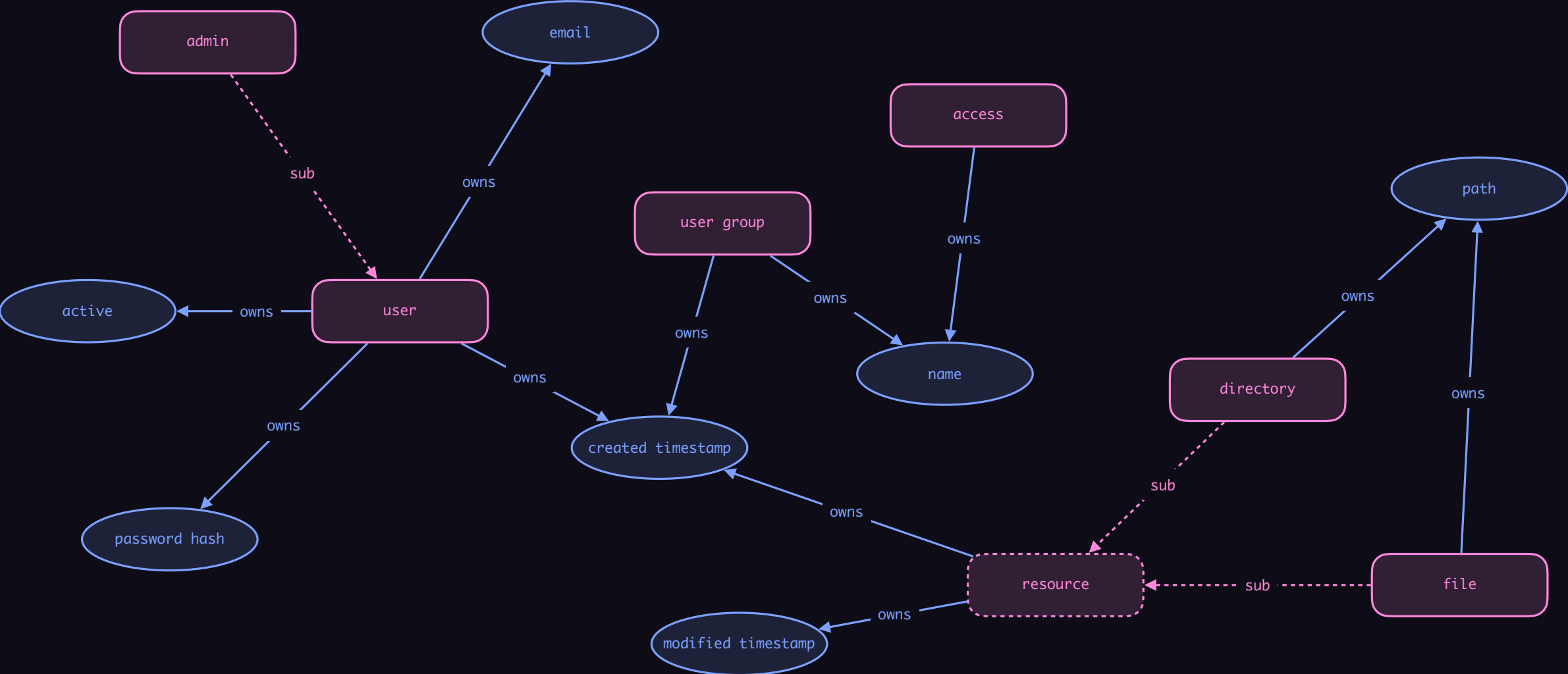
PERA model for a DAC filesystem



PERA model for a DAC filesystem

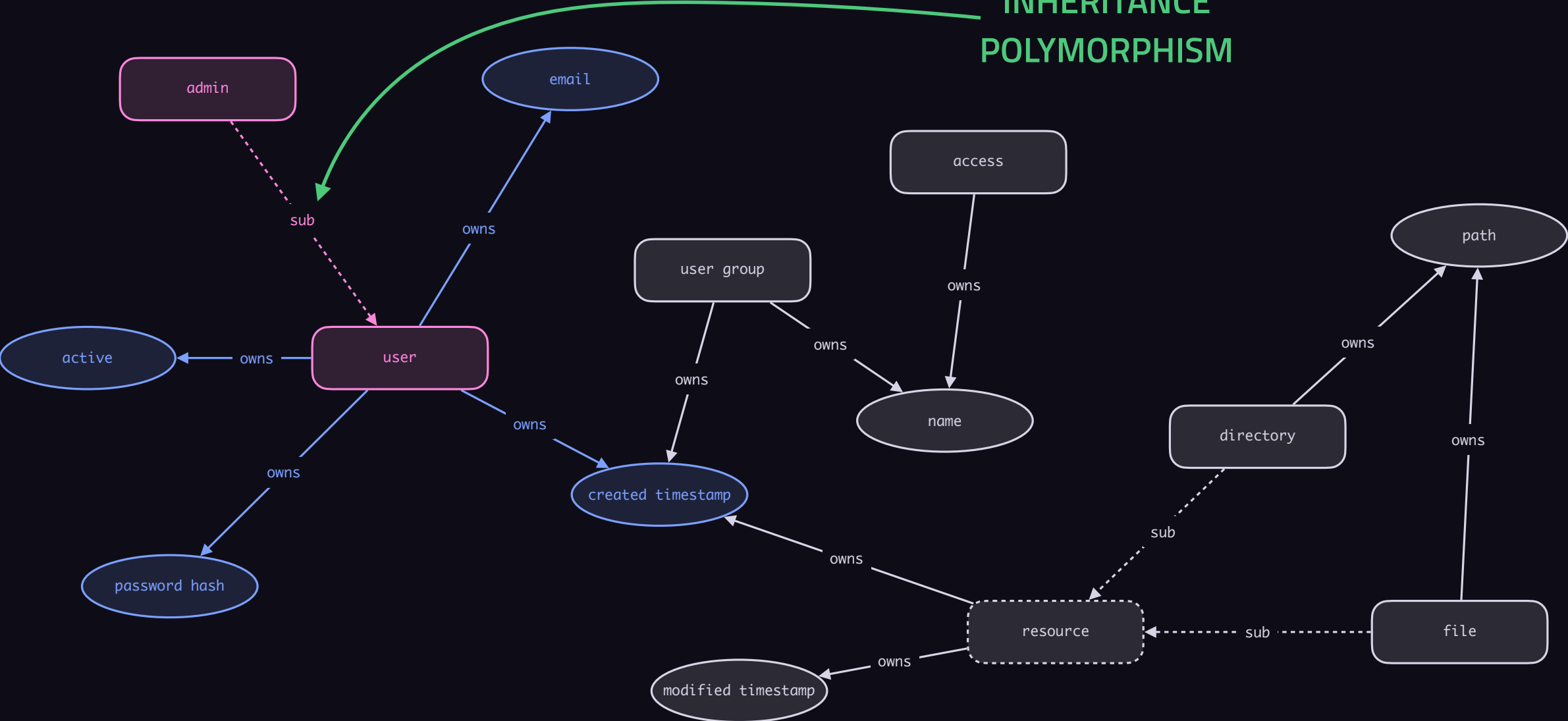


PERA model for a DAC filesystem

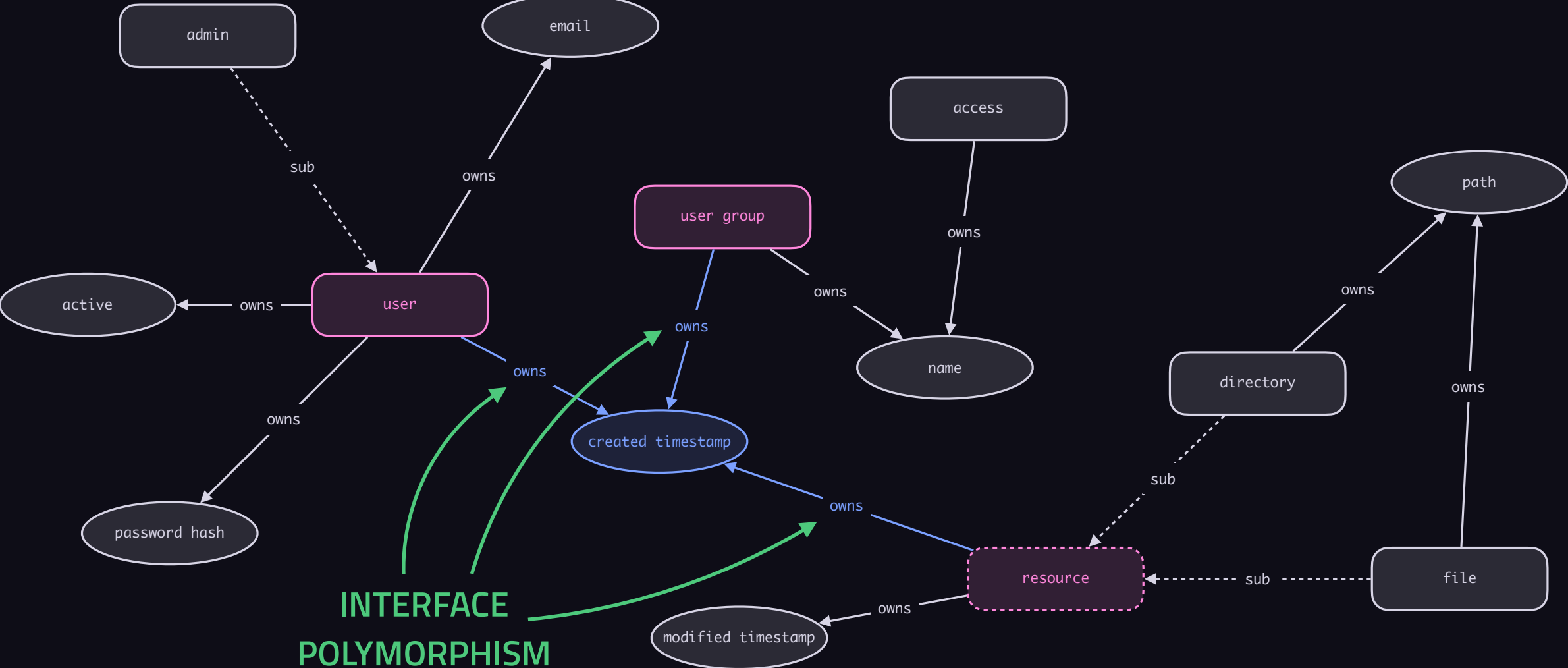


PERA model for a DAC filesystem

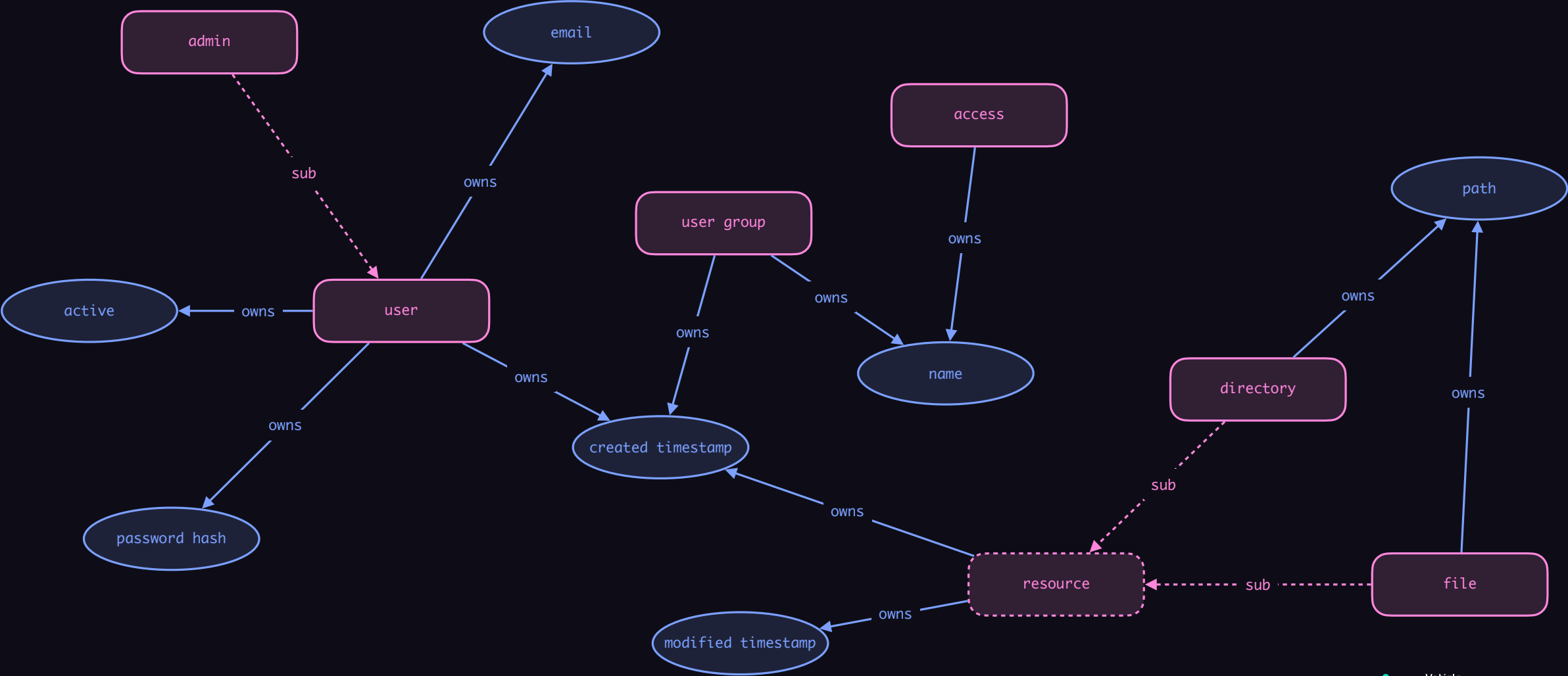
INHERITANCE
POLYMORPHISM



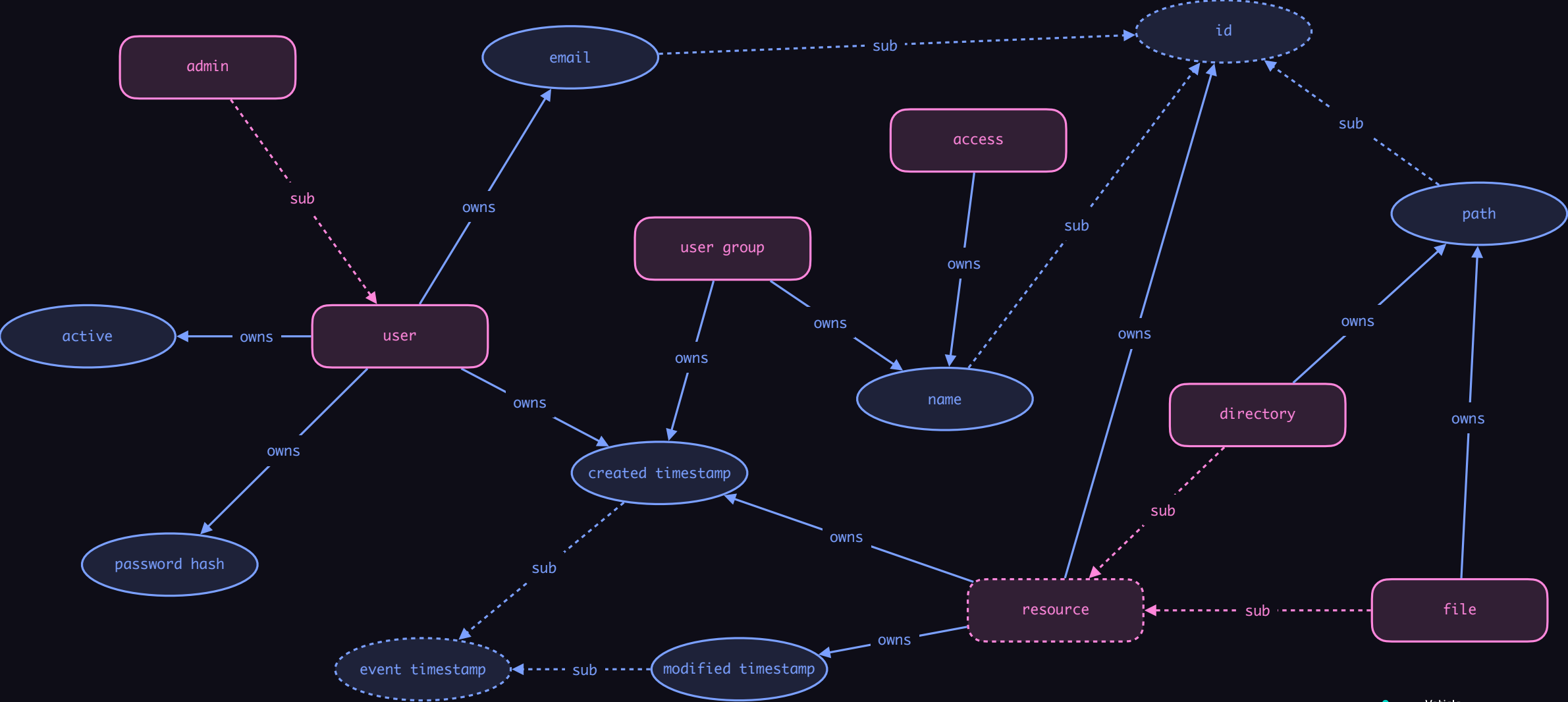
PERA model for a DAC filesystem



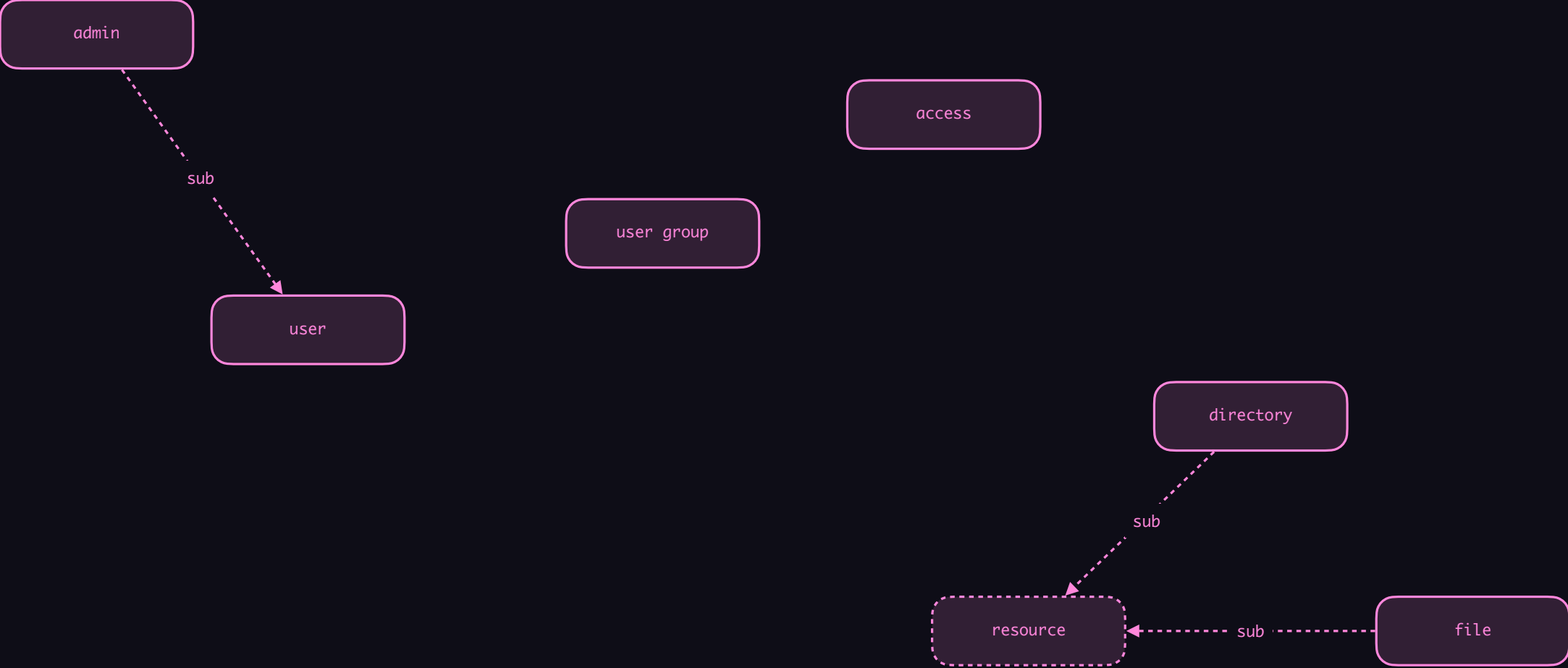
PERA model for a DAC filesystem



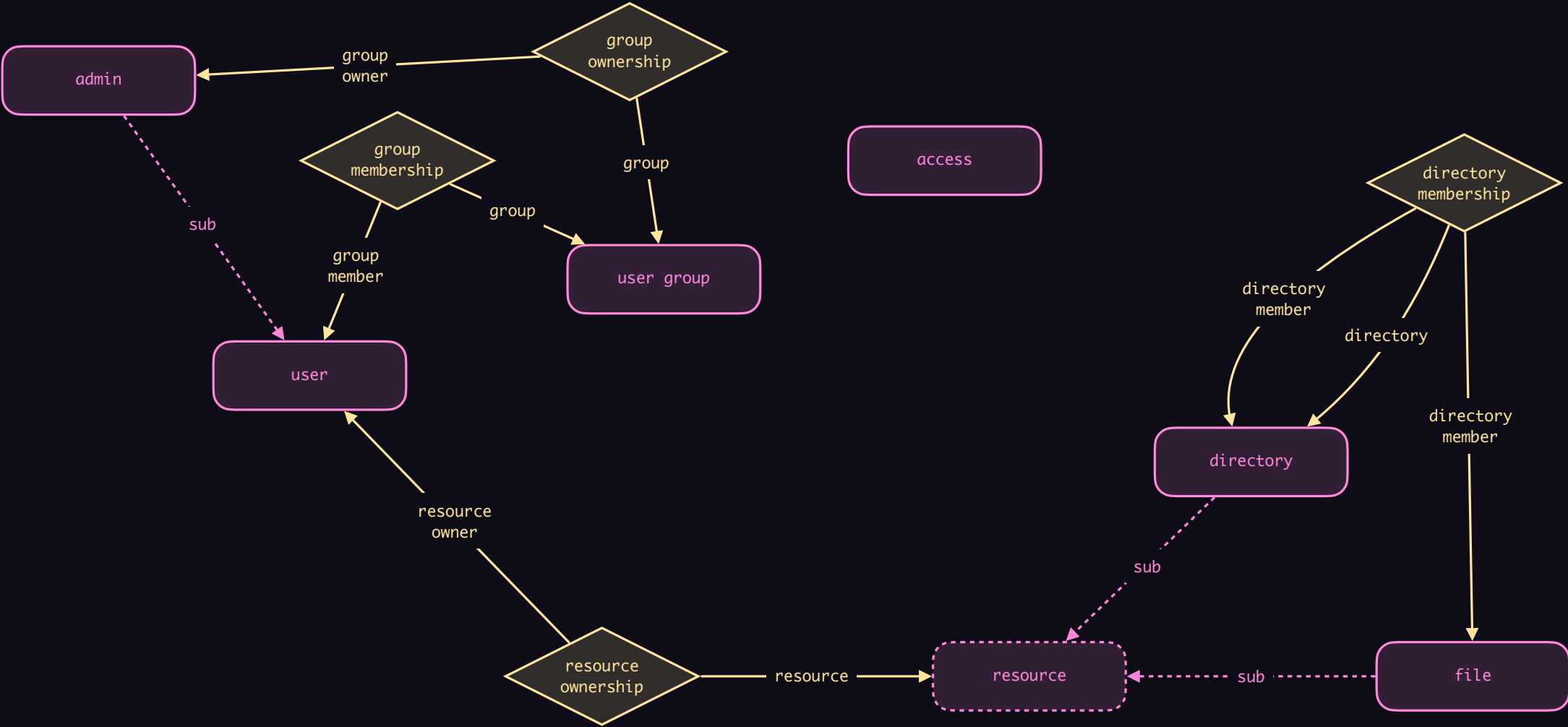
PERA model for a DAC filesystem



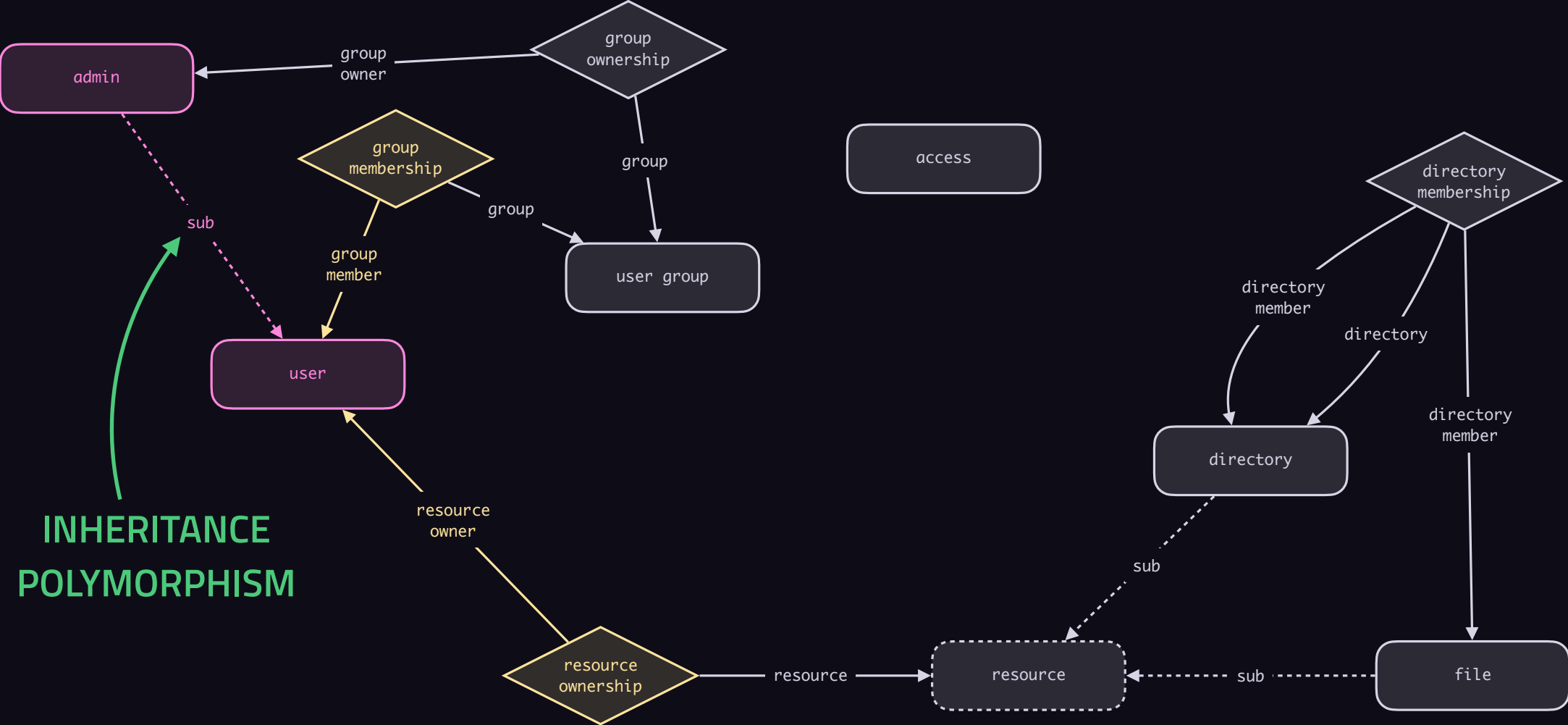
PERA model for a DAC filesystem



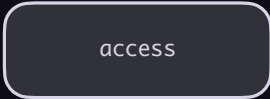
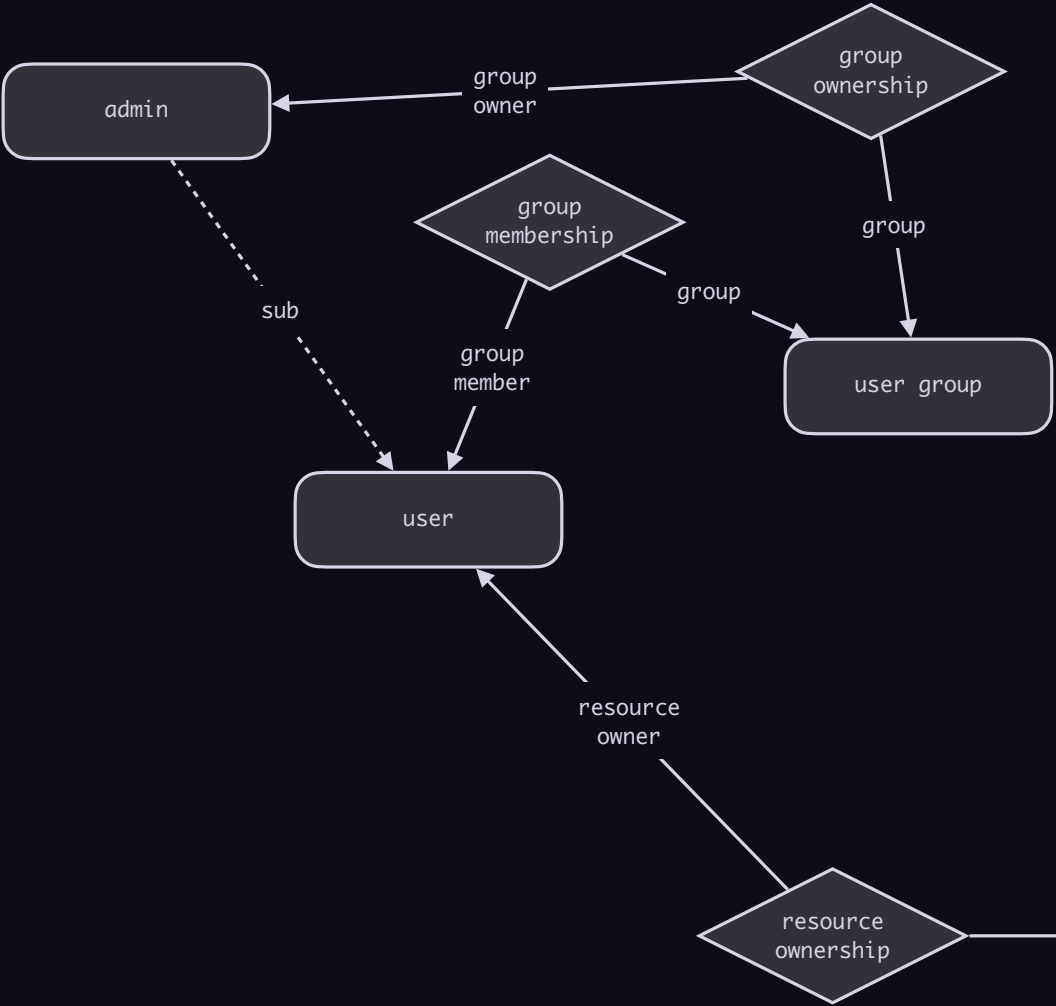
PERA model for a DAC filesystem



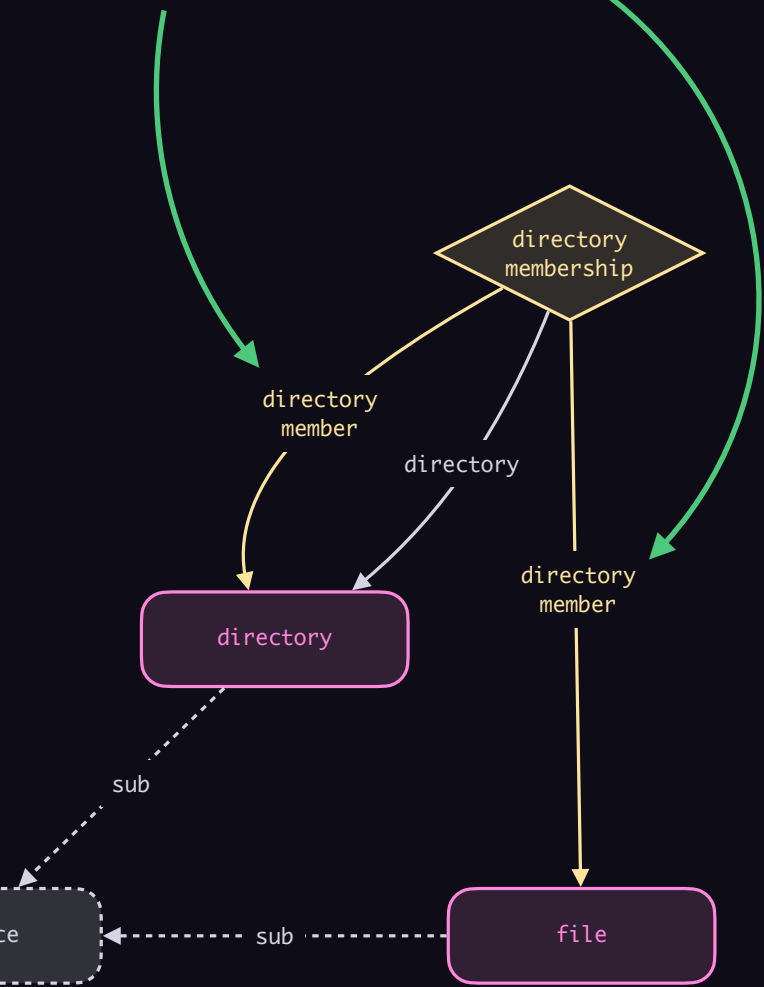
PERA model for a DAC filesystem



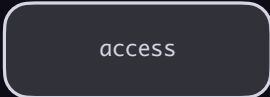
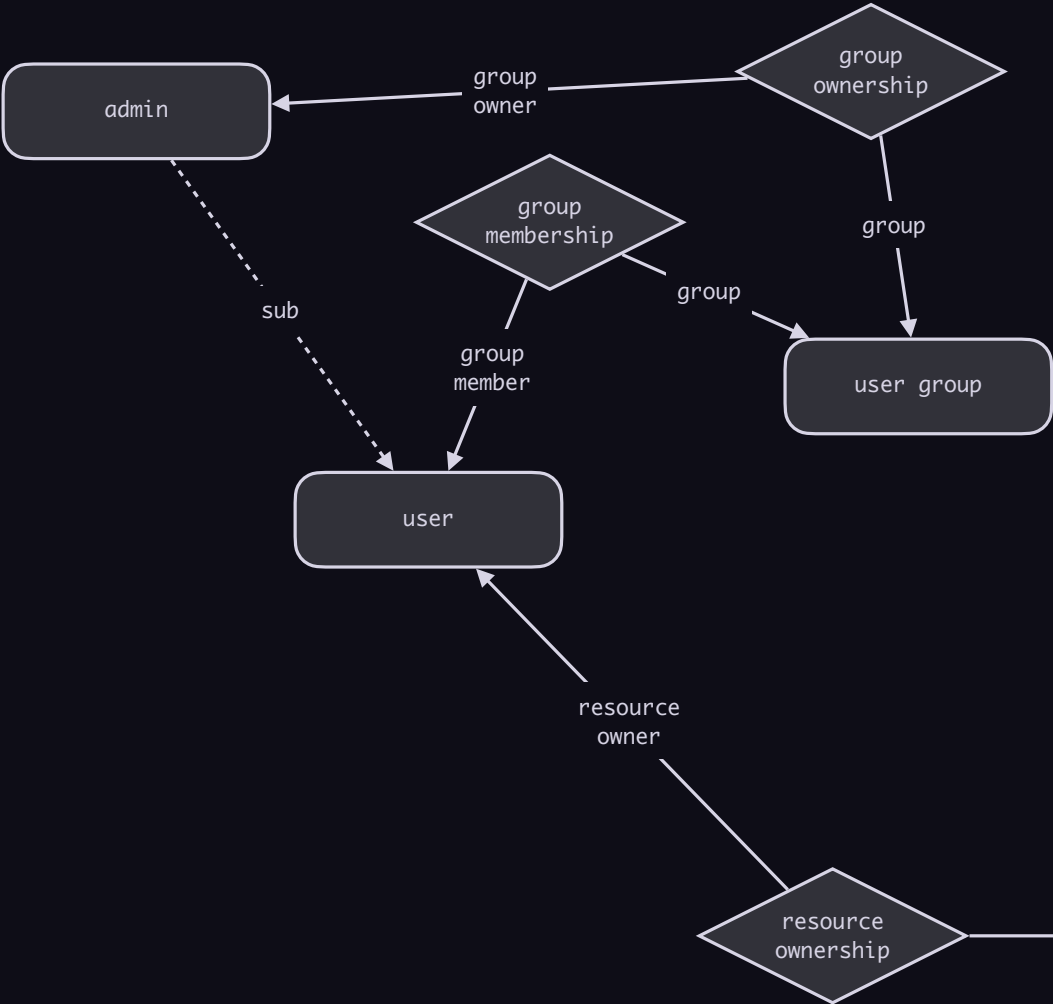
PERA model for a DAC filesystem



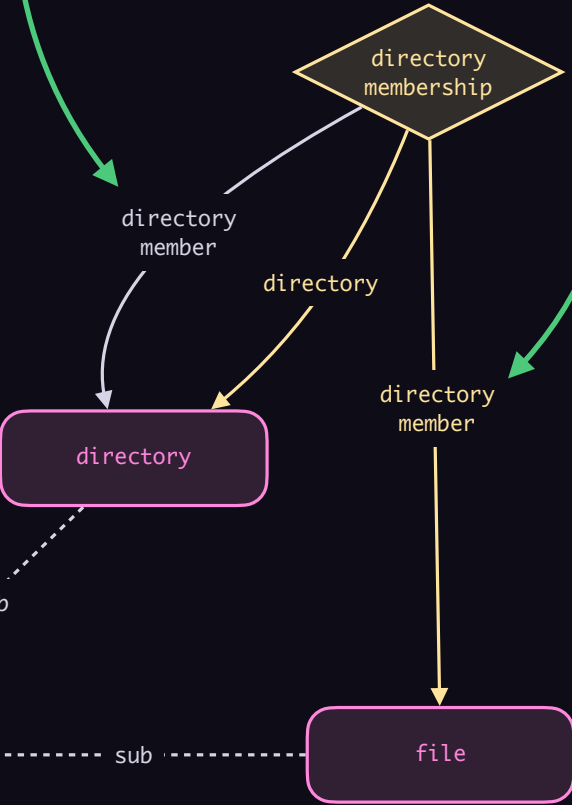
INTERFACE POLYMORPHISM



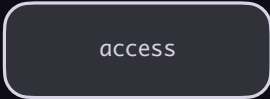
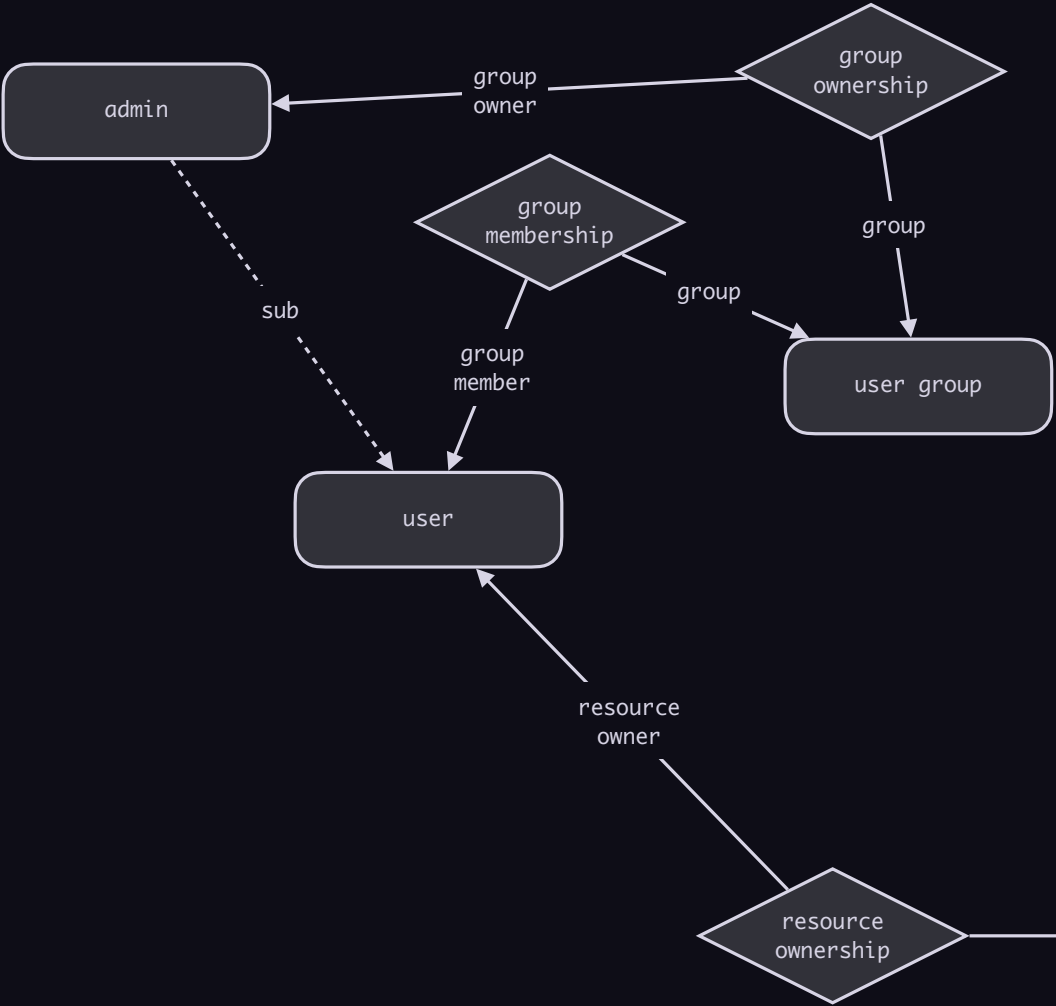
PERA model for a DAC filesystem



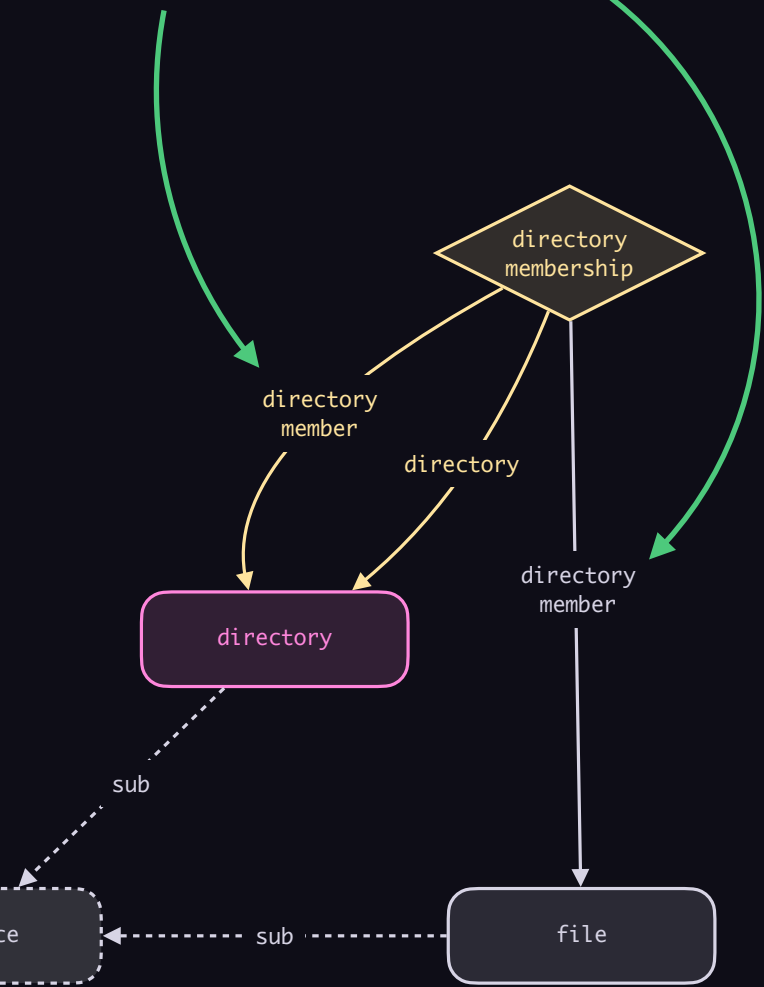
INTERFACE POLYMORPHISM



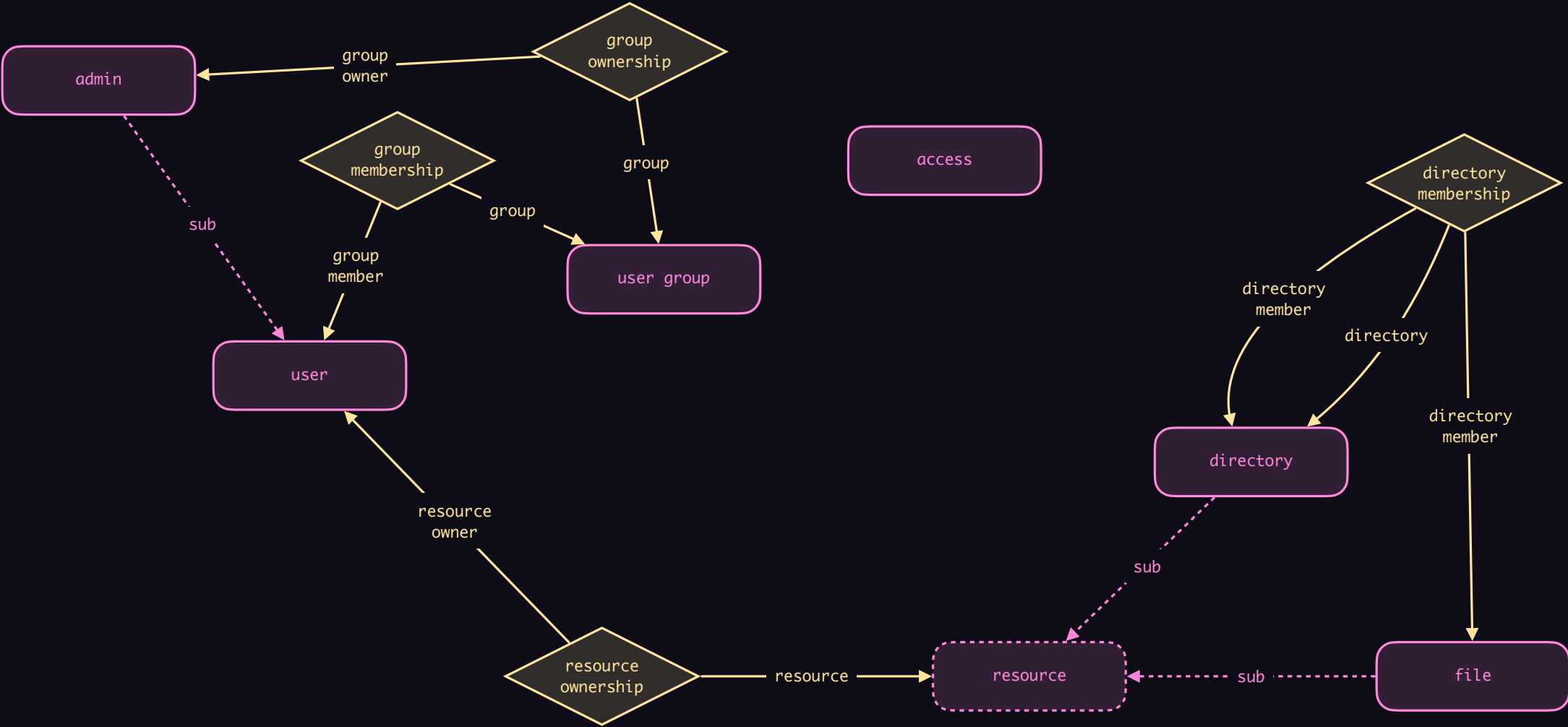
PERA model for a DAC filesystem



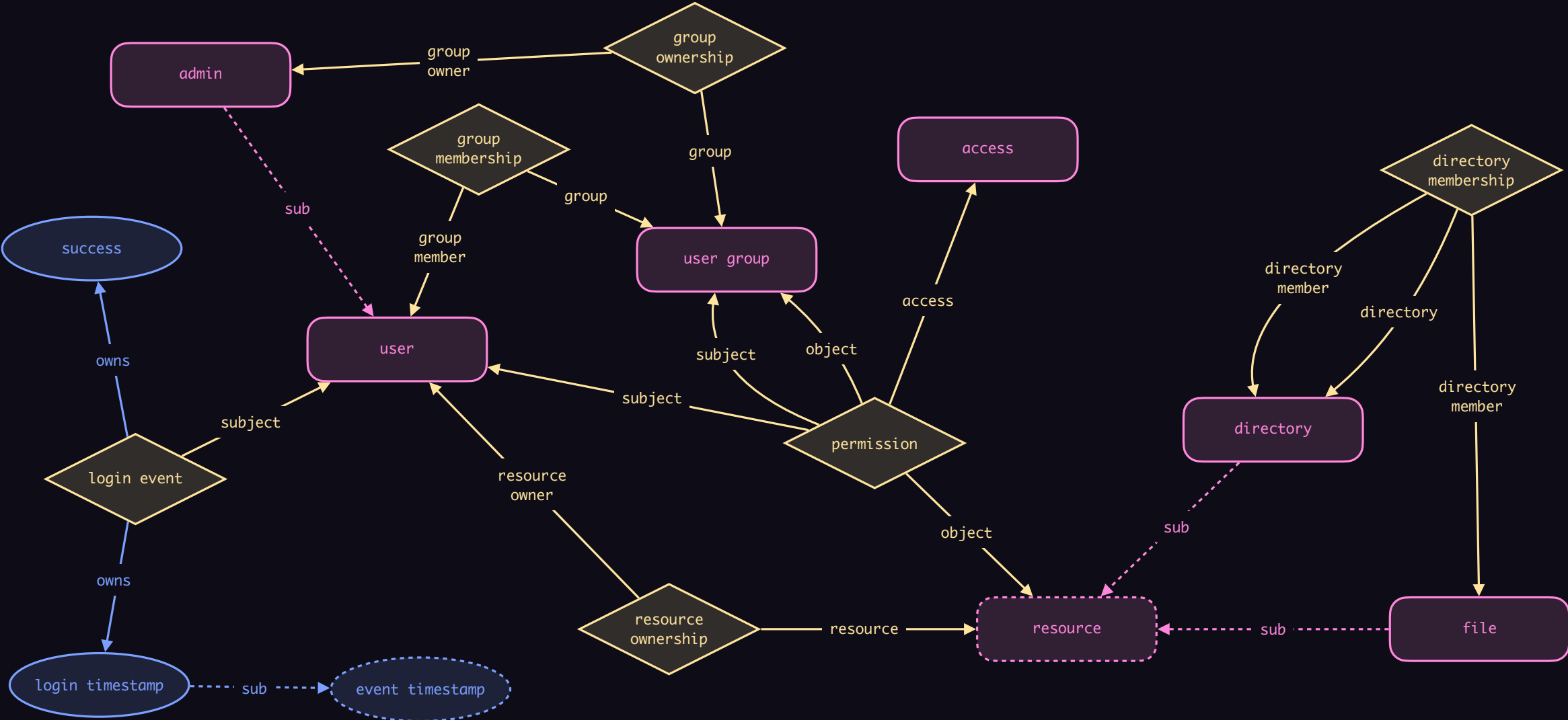
INTERFACE POLYMORPHISM



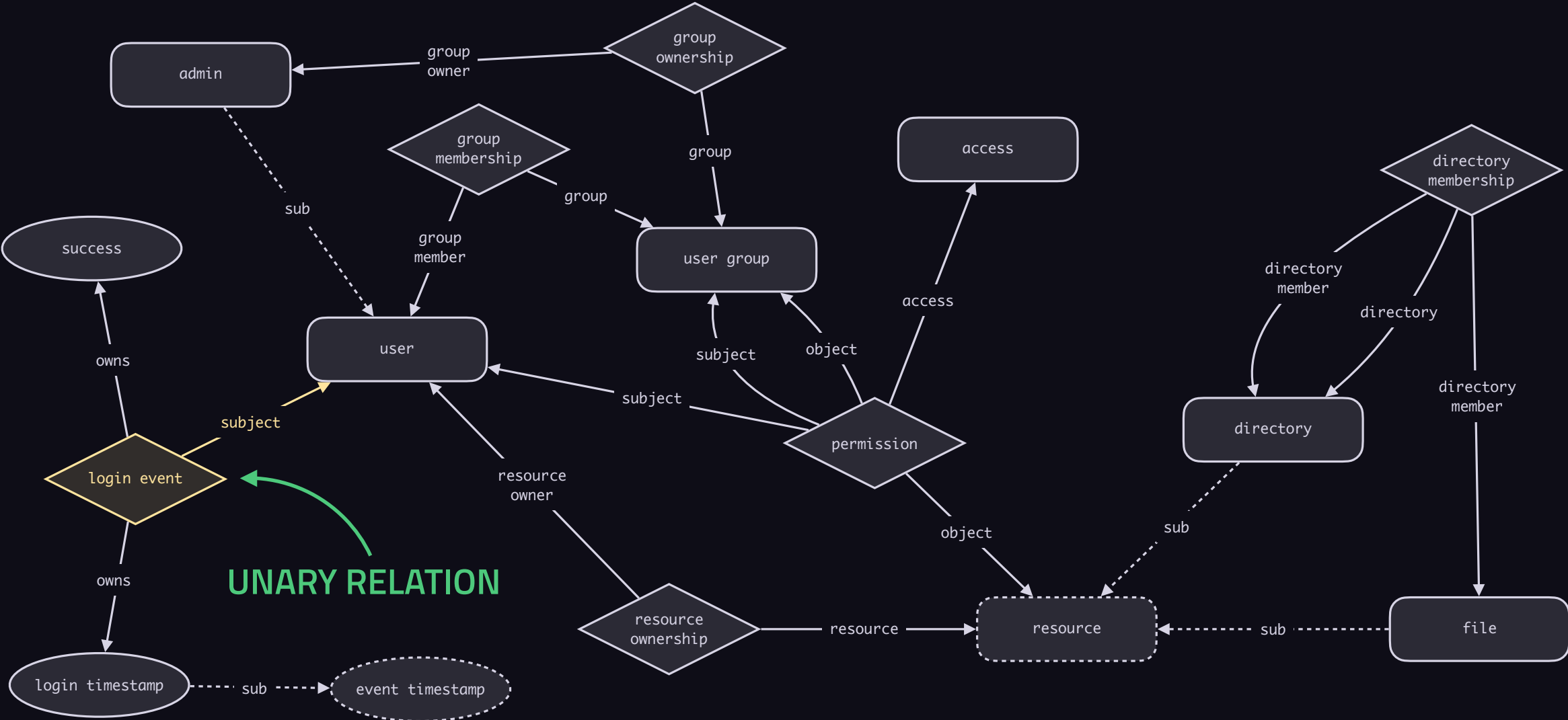
PERA model for a DAC filesystem



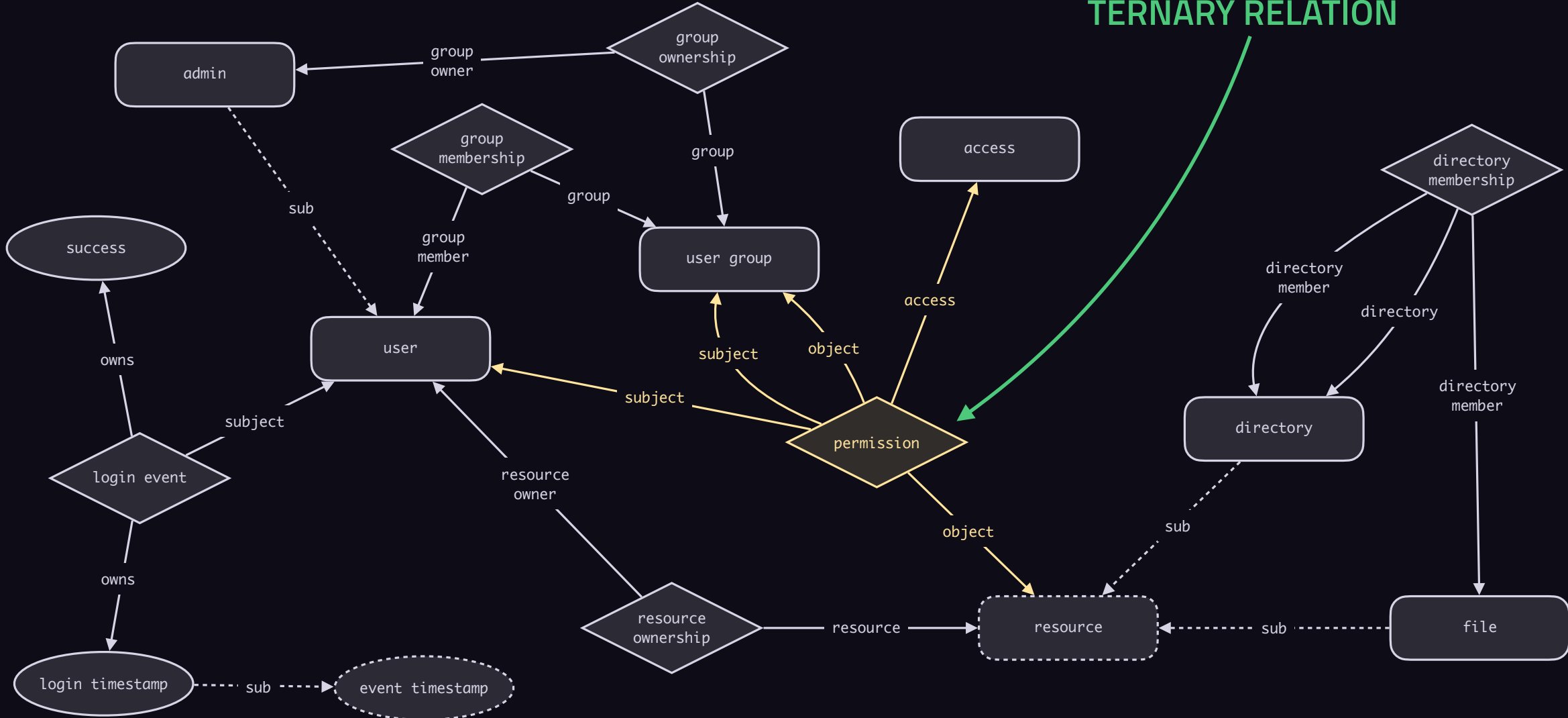
PERA model for a DAC filesystem



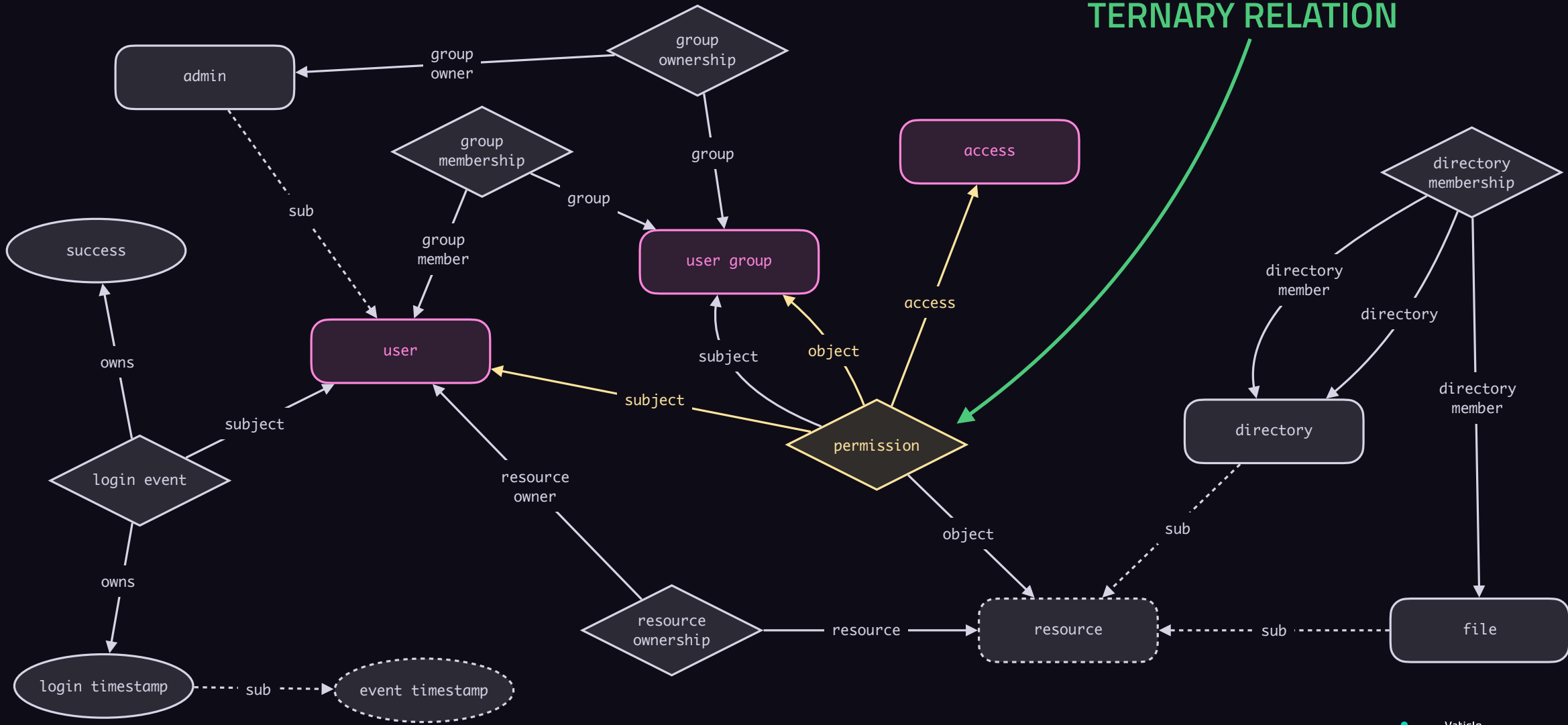
PERA model for a DAC filesystem



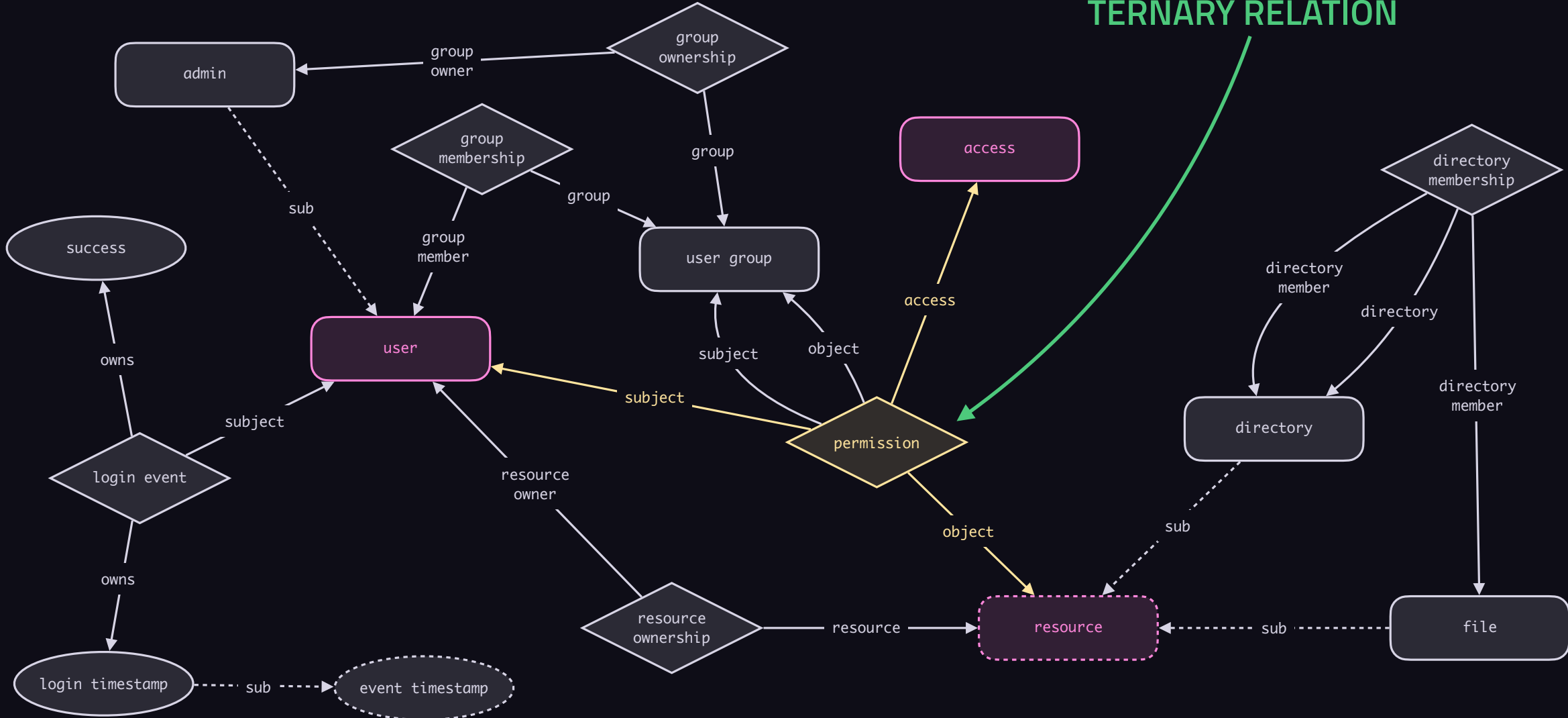
PERA model for a DAC filesystem



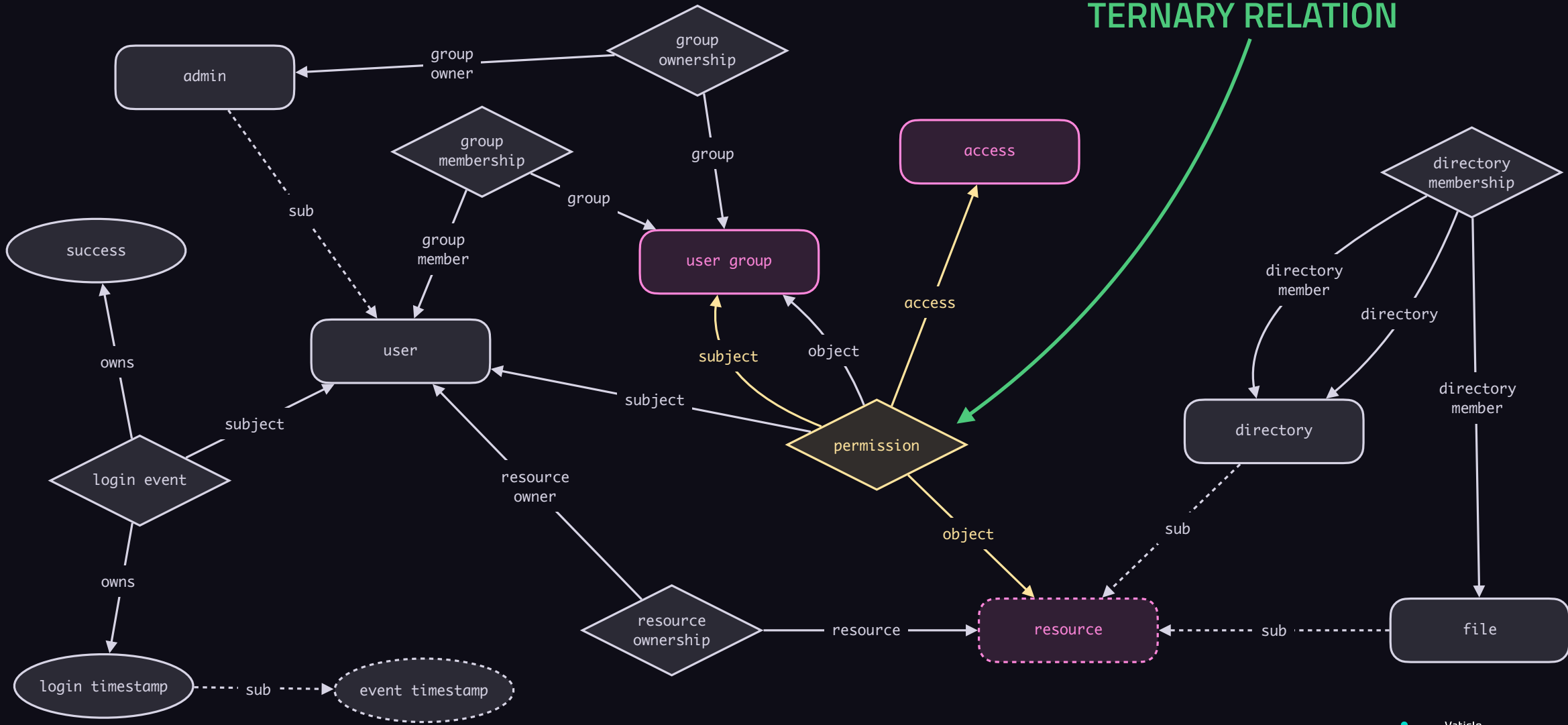
PERA model for a DAC filesystem



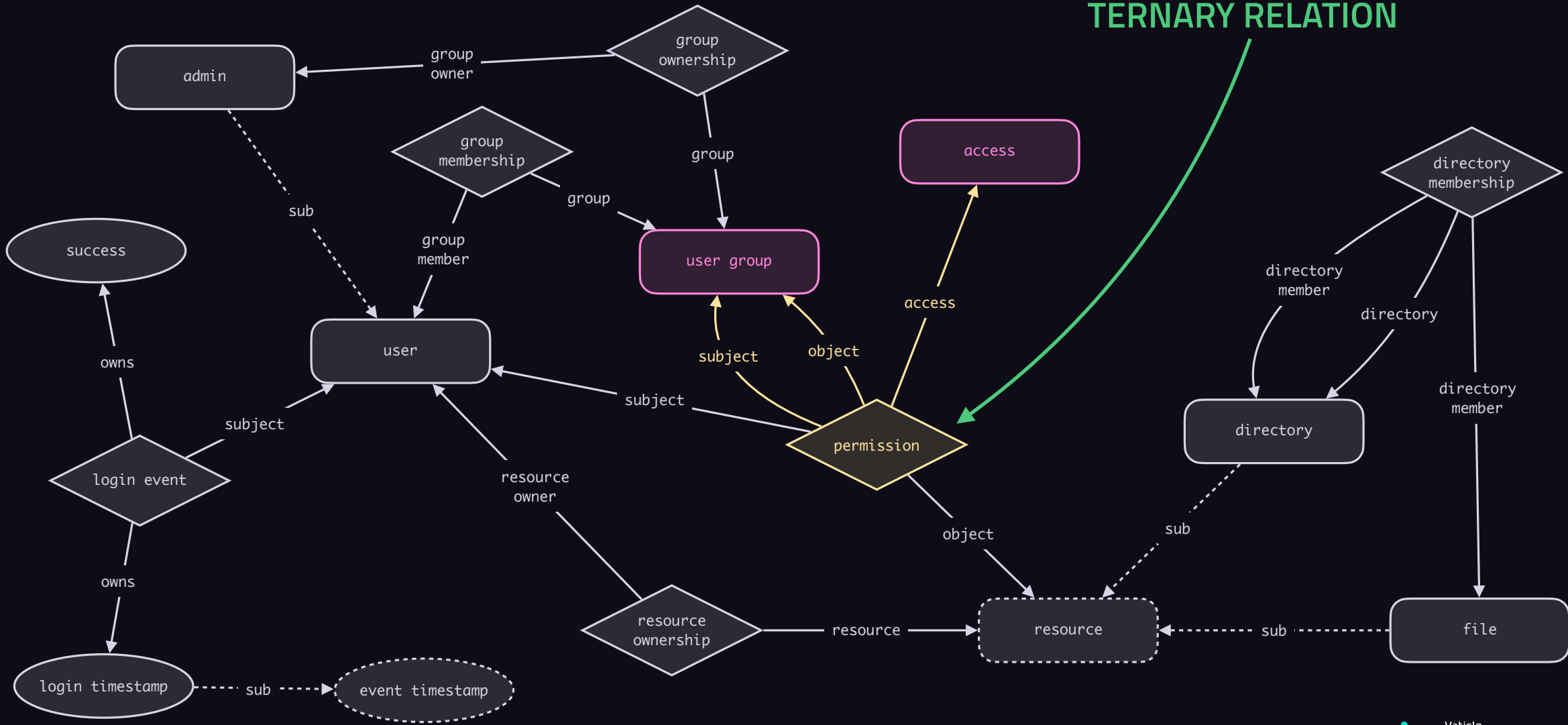
PERA model for a DAC filesystem



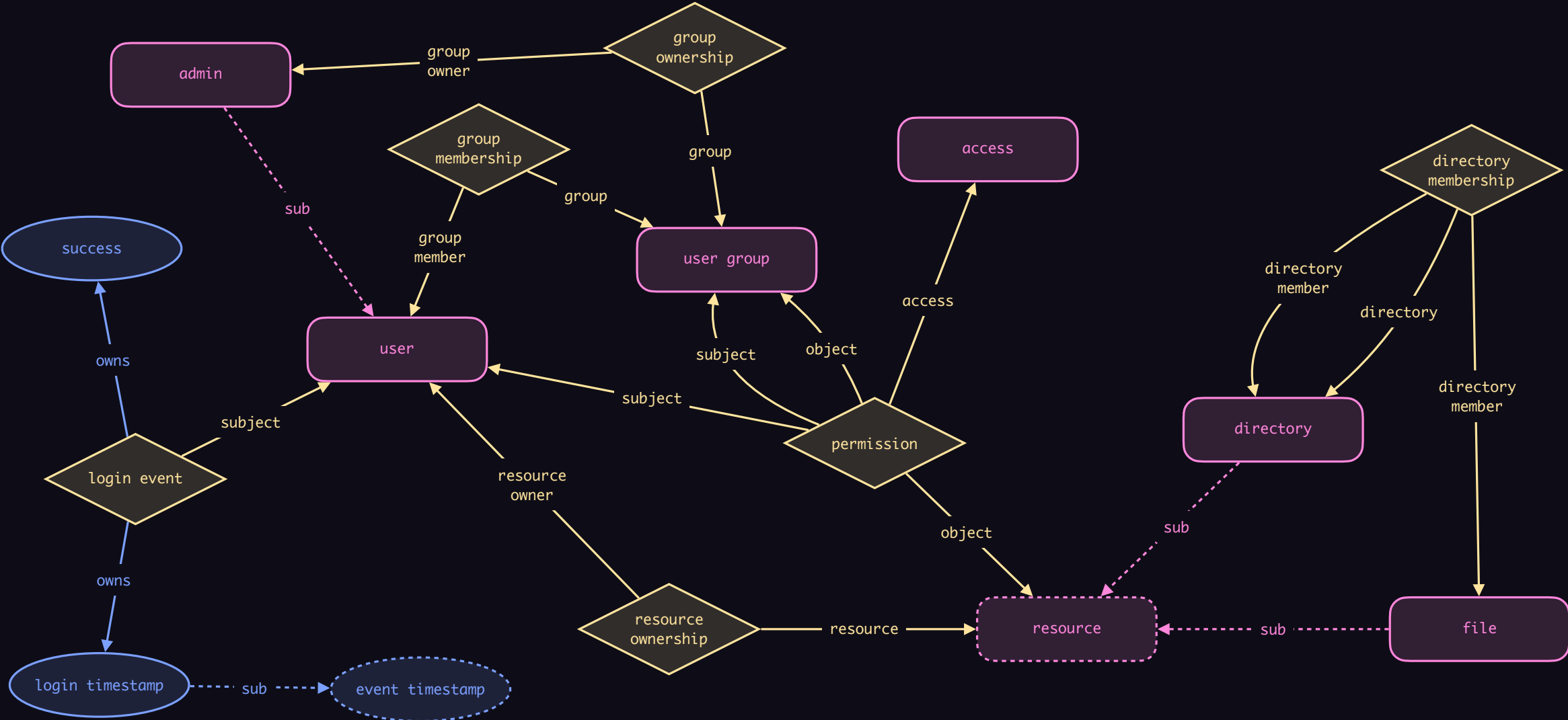
PERA model for a DAC filesystem



PERA model for a DAC filesystem



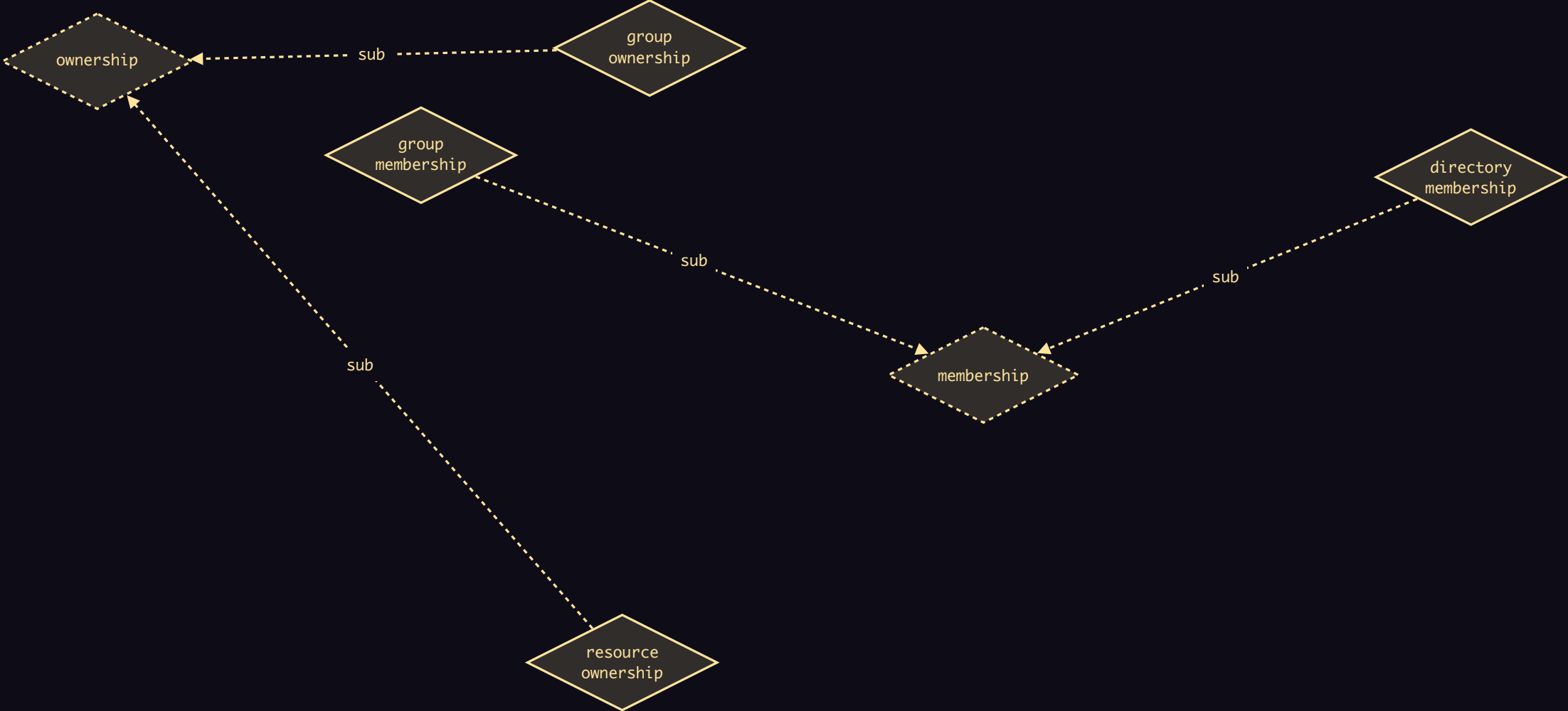
PERA model for a DAC filesystem



PERA model for a DAC filesystem



PERA model for a DAC filesystem



PERA model for a DAC filesystem

Polymorphic features of this model:

- Abstraction.
- Inheritance polymorphism.
- Interface polymorphism.
- N-ary relations.

TypeQL schema for a DAC filesystem

```
define
  user sub entity,
    owns email,
    owns password-hash,
    owns created-timestamp,
    owns active;

  admin sub user;

  user-group sub entity,
    owns name,
    owns created-timestamp;

  resource sub entity, abstract,
    owns id,
    owns created-timestamp,
    owns modified-timestamp;

  file sub resource,
    owns path as id;

  directory sub resource,
    owns path as id;

  access sub entity,
    owns name;

  id sub attribute, abstract, value string;
  email sub id;
  name sub id;
  path sub id;
  password-hash sub attribute, value string;
  event-timestamp sub attribute, abstract, value datetime;
  created-timestamp sub event-timestamp;
  modified-timestamp sub event-timestamp;
  active sub attribute, value boolean;
```

TypeQL schema for a DAC filesystem

```
define

user sub entity,
  owns email,
  owns password-hash,
  owns created-timestamp,
  owns active;

admin sub user;

user-group sub entity,
  owns name,
  owns created-timestamp;

resource sub entity, abstract,
  owns id,
  owns created-timestamp,
  owns modified-timestamp;

file sub resource,
  owns path as id;

directory sub resource,
  owns path as id;

access sub entity,
  owns name;

id sub attribute, abstract, value string;
email sub id;
name sub id;
path sub id;
password-hash sub attribute, value string;
event-timestamp sub attribute, abstract, value datetime;
created-timestamp sub event-timestamp;
modified-timestamp sub event-timestamp;
active sub attribute, value boolean;
```

ABSTRACTION



TypeQL schema for a DAC filesystem

INHERITANCE POLYMORPHISM

```
define
```

```
user sub entity,  
  owns email,  
  owns password-hash,  
  owns created-timestamp,  
  owns active;
```

```
admin sub user;
```

```
user-group sub entity,  
  owns name,  
  owns created-timestamp;
```

```
resource sub entity, abstract,  
  owns id,  
  owns created-timestamp,  
  owns modified-timestamp;
```

```
file sub resource,  
  owns path as id;
```

```
directory sub resource,  
  owns path as id;
```

```
access sub entity,  
  owns name;
```

```
id sub attribute, abstract, value string;  
email sub id;  
name sub id;  
path sub id;  
password-hash sub attribute, value string;  
event-timestamp sub attribute, abstract, value datetime;  
created-timestamp sub event-timestamp;  
modified-timestamp sub event-timestamp;  
active sub attribute, value boolean;
```

TypeQL schema for a DAC filesystem

```
define
  user sub entity,
    owns email,
    owns password-hash,
    owns created-timestamp,
    owns active;

  admin sub user;

  user-group sub entity,
    owns name,
    owns created-timestamp;

  resource sub entity, abstract,
    owns id,
    owns created-timestamp,
    owns modified-timestamp;
```

```
file sub resource,
  owns path as id;

directory sub resource,
  owns path as id;

access sub entity,
  owns name;

id sub attribute, abstract, value string;
email sub id;
name sub id;
path sub id;
password-hash sub attribute, value string;
event-timestamp sub attribute, abstract, value datetime;
created-timestamp sub event-timestamp;
modified-timestamp sub event-timestamp;
active sub attribute, value boolean;
```

**INTERFACE
POLYMORPHISM**

TypeQL schema for a DAC filesystem

```
define
```

```
ownership sub relation, abstract,  
  relates owned,  
  relates owner;
```

```
group-ownership sub ownership,  
  relates group as owned,  
  relates group-owner as owner;
```

```
resource-ownership sub ownership,  
  relates resource as owned,  
  relates resource-owner as owner;
```

```
membership sub relation, abstract,  
  relates parent,  
  relates member;
```

```
group-membership sub membership,  
  relates group as parent,  
  relates group-member as member;
```

```
directory-membership sub membership,  
  relates directory as parent,  
  relates directory-member as member;
```

```
permission sub relation,  
  relates subject,  
  relates object,  
  relates access;
```

```
login-event sub relation,  
  relates subject,  
  owns login-timestamp,  
  owns success;
```

```
login-timestamp sub event-timestamp;  
success sub attribute, value boolean;
```

TypeQL schema for a DAC filesystem

```
define
```

```
ownership sub relation, abstract,  
  relates owned,  
  relates owner;
```

```
group-ownership sub ownership,  
  relates group as owned,  
  relates group-owner as owner;
```

```
resource-ownership sub ownership,  
  relates resource as owned,  
  relates resource-owner as owner;
```

```
membership sub relation, abstract,  
  relates parent,  
  relates member;
```

```
group-membership sub membership,  
  relates group as parent,  
  relates group-member as member;
```

```
directory-membership sub membership,  
  relates directory as parent,  
  relates directory-member as member;
```

```
permission sub relation,  
  relates subject,  
  relates object,  
  relates access;
```

```
login-event sub relation,  
  relates subject,  
  owns login-timestamp,  
  owns success;
```

```
login-timestamp sub event-timestamp;  
success sub attribute, value boolean;
```

UNARY RELATION



TypeQL schema for a DAC filesystem

TERNARY RELATION

```
define
```

```
ownership sub relation, abstract,  
  relates owned,  
  relates owner;
```

```
group-ownership sub ownership,  
  relates group as owned,  
  relates group-owner as owner;
```

```
resource-ownership sub ownership,  
  relates resource as owned,  
  relates resource-owner as owner;
```

```
membership sub relation, abstract,  
  relates parent,  
  relates member;
```

```
group-membership sub membership,  
  relates group as parent,  
  relates group-member as member;
```

```
directory-membership sub membership,  
  relates directory as parent,  
  relates directory-member as member;
```

```
permission sub relation,  
  relates subject,  
  relates object,  
  relates access;
```

```
login-event sub relation,  
  relates subject,  
  owns login-timestamp,  
  owns success;
```

```
login-timestamp sub event-timestamp;  
success sub attribute, value boolean;
```

TypeQL schema for a DAC filesystem

```
define

user plays resource-ownership:resource-owner,
      plays group-membership:group-member,
      plays permission:subject,
      plays login-event:subject;

admin plays group-ownership:group-owner;

user-group plays group-ownership:group,
            plays resource-ownership:resource-owner,
            plays group-membership:group,
            plays permission:subject,
            plays permission:object;

resource plays resource-ownership:resource,
            plays permission:object;

file plays directory-membership:directory-member;

directory plays directory-membership:directory,
            plays directory-membership:directory-member;

access plays permission:access;
```


TypeQL schema for a DAC filesystem

```
define

user plays resource-ownership:resource-owner,
    plays group-membership:group-member,
    plays permission:subject,
    plays login-event:subject;

admin plays group-ownership:group-owner;

user-group plays group-ownership:group,
    plays resource-ownership:resource-owner,
    plays group-membership:group,
    plays permission:subject,
    plays permission:object;

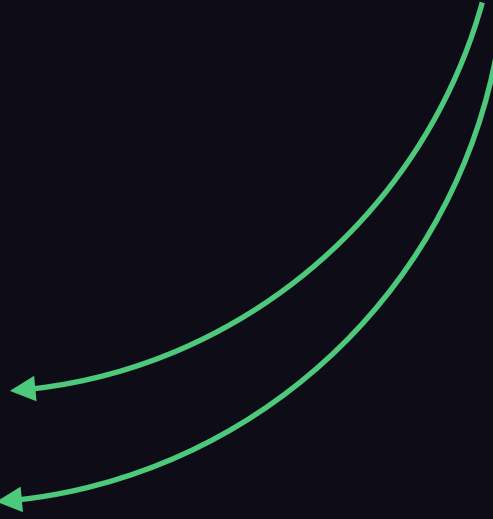
resource plays resource-ownership:resource,
    plays permission:object;

file plays directory-membership:directory-member;

directory plays directory-membership:directory,
    plays directory-membership:directory-member;

access plays permission:access;
```

INTERFACE
POLYMORPHISM



Semantic integrity of inserted data

```
insert
$rhonda isa user,
  has email "rhonda@vaticle.com",
  has active true,
  has created-timestamp 2023-11-10T15:19:43,
  has password-hash "6f1127d4b8ee9bd64df9b0ae3f8a7f58";
```

```
insert
$cedric isa admin,
  has email "cedric@vaticle.com",
  has active true,
  has created-timestamp 2023-01-01T00:00:00,
  has password-hash "e0d29e328f65b8074d7df218c73b1726";
```

```
match
$rhonda isa user, has email "rhonda@vaticle.com";
insert
(subject: $rhonda) isa login-event,
  has login-timestamp 2023-12-29T13:38:51,
  has success true;
```

Semantic integrity of inserted data

```
insert
$rhonda isa user,
  has email "rhonda@vaticle.com",
  has active true,
  has created-timestamp 2023-11-10T15:19:43,
  has password-hash "6f1127d4b8ee9bd64df9b0ae3f8a7f58";
```

Valid: user owns email;
Valid: user owns active;
Valid: user owns created-timestamp;
Valid: user owns password-hash;

```
insert
$cedric isa admin,
  has email "cedric@vaticle.com",
  has active true,
  has created-timestamp 2023-01-01T00:00:00,
  has password-hash "e0d29e328f65b8074d7df218c73b1726";
```

Valid: admin sub user; user owns email;
Valid: admin sub user; user owns active;
Valid: admin sub user; user owns created-timestamp;
Valid: admin sub user; user owns password-hash;

```
match
$rhonda isa user, has email "rhonda@vaticle.com";
insert
(subject: $rhonda) isa login-event,
  has login-timestamp 2023-12-29T13:38:51,
  has success true;
```

Valid: login-event relates subject; user plays login-event:subject;
Valid: login-event owns login-timestamp;
Valid: login-event owns success;

Semantic integrity of inserted data

```
insert
$omar isa user,
  has path "/vaticle/omar",
  has success true;
```

```
insert
$avon isa user, has email "avon@vaticle.com";
$researchers isa user-group, has name "researchers";
(group: $researchers, group-owner: $avon) isa group-ownership;
```

Semantic integrity of inserted data

```
insert
$omar isa user,
  has path "/vaticle/omar",
  has success true;
```

```
## Error> [CXN05] The transaction is closed because of the error(s):
[THW03] Invalid Write: Attribute of type 'success' is not defined to be owned by type 'user'.
## Terminated
```

```
insert
$avon isa user, has email "avon@vaticle.com";
$researchers isa user-group, has name "researchers";
(group: $researchers, group-owner: $avon) isa group-ownership;
```

```
## Error> [CXN05] The transaction is closed because of the error(s):
[THW08] Invalid Write: The type 'user' does not play the role type 'group-ownership:group-owner'.
## Terminated
```

Conceptual model

- PERA model captures polymorphism in data.
- TypeQL schema directly implements model without mismatch.
- Inserted data is semantically validated against the schema.



Type-theoretic language

Declarative polymorphic querying

Types of **polymorphic querying**

- Querying with inheritance polymorphism.
- Querying with interface polymorphism.
- Querying with parametric polymorphism.
- Querying the schema.

Querying with **inheritance** polymorphism

"List the email and active status for each user."

```
match
$user isa user;
fetch
$user: email, active;
```

Querying with inheritance polymorphism

"List the email and active status for each user."

```
{
  "user": {
    "active": [ { "value": true, "value_type": "boolean", "type": { "label": "active", "root": "attribute" } } ],
    "email": [ { "value": "cedric@vaticle.com", "value_type": "string", "type": { "label": "email", "root": "attribute" } } ],
    "type": { "label": "admin", "root": "entity" }
  }
}
{
  "user": {
    "active": [ { "value": true, "value_type": "boolean", "type": { "label": "active", "root": "attribute" } } ],
    "email": [ { "value": "reginald@vaticle.com", "value_type": "string", "type": { "label": "email", "root": "attribute" } } ],
    "type": { "label": "user", "root": "entity" }
  }
}
{
  "user": {
    "active": [ { "value": true, "value_type": "boolean", "type": { "label": "active", "root": "attribute" } } ],
    "email": [ { "value": "tommy@vaticle.com", "value_type": "string", "type": { "label": "email", "root": "attribute" } } ],
    "type": { "label": "user", "root": "entity" }
  }
}
```

Querying with inheritance polymorphism

"List the email and active status for each user."

```
{
  "user": {
    "active": [ { "value": true, "value_type": "boolean", "type": { "label": "active", "root": "attribute" } } ],
    "email": [ { "value": "cedric@vaticle.com", "value_type": "string", "type": { "label": "email", "root": "attribute" } } ],
    "type": { "label": "admin", "root": "entity" }
  }
}
{
  "user": {
    "active": [ { "value": true, "value_type": "boolean", "type": { "label": "active", "root": "attribute" } } ],
    "email": [ { "value": "reginald@vaticle.com", "value_type": "string", "type": { "label": "email", "root": "attribute" } } ],
    "type": { "label": "user", "root": "entity" }
  }
}
{
  "user": {
    "active": [ { "value": true, "value_type": "boolean", "type": { "label": "active", "root": "attribute" } } ],
    "email": [ { "value": "tommy@vaticle.com", "value_type": "string", "type": { "label": "email", "root": "attribute" } } ],
    "type": { "label": "user", "root": "entity" }
  }
}
```

The type of `$user`

Querying with inheritance polymorphism

"List the email and active status for each user."

```
{
  "user": {
    "active": [ { "value": true, "value_type": "boolean", "type": { "label": "active", "root": "attribute" } } ],
    "email": [ { "value": "cedric@vaticle.com", "value_type": "string", "type": { "label": "email", "root": "attribute" } } ],
    "type": { "label": "admin", "root": "entity" }
  }
}
{
  "user": {
    "active": [ { "value": true, "value_type": "boolean", "type": { "label": "active", "root": "attribute" } } ],
    "email": [ { "value": "reginald@vaticle.com", "value_type": "string", "type": { "label": "email", "root": "attribute" } } ],
    "type": { "label": "user", "root": "entity" }
  }
}
{
  "user": {
    "active": [ { "value": true, "value_type": "boolean", "type": { "label": "active", "root": "attribute" } } ],
    "email": [ { "value": "tommy@vaticle.com", "value_type": "string", "type": { "label": "email", "root": "attribute" } } ],
    "type": { "label": "user", "root": "entity" }
  }
}
```

The email of `$user`

Querying with inheritance polymorphism

"List the email and active status for each user."

```
{
  "user": {
    "active": [ { "value": true, "value_type": "boolean", "type": { "label": "active", "root": "attribute" } } ],
    "email": [ { "value": "cedric@vaticle.com", "value_type": "string", "type": { "label": "email", "root": "attribute" } } ],
    "type": { "label": "admin", "root": "entity" }
  }
}
{
  "user": {
    "active": [ { "value": true, "value_type": "boolean", "type": { "label": "active", "root": "attribute" } } ],
    "email": [ { "value": "reginald@vaticle.com", "value_type": "string", "type": { "label": "email", "root": "attribute" } } ],
    "type": { "label": "user", "root": "entity" }
  }
}
{
  "user": {
    "active": [ { "value": true, "value_type": "boolean", "type": { "label": "active", "root": "attribute" } } ],
    "email": [ { "value": "tommy@vaticle.com", "value_type": "string", "type": { "label": "email", "root": "attribute" } } ],
    "type": { "label": "user", "root": "entity" }
  }
}
```

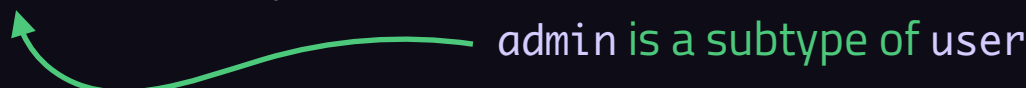
The active status of `$user`

Querying with inheritance polymorphism

"List the email and active status for each user."

```
{
  "user": {
    "active": [ { "value": true, "value_type": "boolean", "type": { "label": "active", "root": "attribute" } } ],
    "email": [ { "value": "cedric@vaticle.com", "value_type": "string", "type": { "label": "email", "root": "attribute" } } ],
    "type": { "label": "admin", "root": "entity" }
  }
}
{
  "user": {
    "active": [ { "value": true, "value_type": "boolean", "type": { "label": "active", "root": "attribute" } } ],
    "email": [ { "value": "reginald@vaticle.com", "value_type": "string", "type": { "label": "email", "root": "attribute" } } ],
    "type": { "label": "user", "root": "entity" }
  }
}
{
  "user": {
    "active": [ { "value": true, "value_type": "boolean", "type": { "label": "active", "root": "attribute" } } ],
    "email": [ { "value": "tommy@vaticle.com", "value_type": "string", "type": { "label": "email", "root": "attribute" } } ],
    "type": { "label": "user", "root": "entity" }
  }
}
```

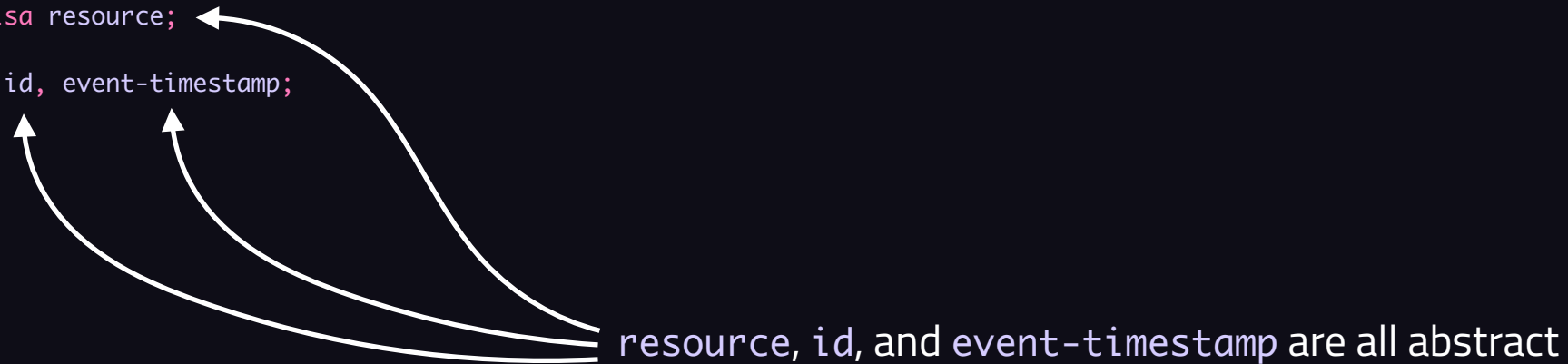
admin is a subtype of user



Querying with inheritance polymorphism

"List the ID and event timestamps of every resource."

```
match  
$resource isa resource;  
fetch  
$resource: id, event-timestamp;
```



Querying with inheritance polymorphism

"List the ID and event timestamps of every resource."

```
{
  "resource": {
    "event-timestamp": [
      { "value": "2023-08-08T11:57:54.000", "value_type": "datetime", "type": { "label": "modified-timestamp", "root": "attribute" } },
      { "value": "2023-08-13T13:16:10.000", "value_type": "datetime", "type": { "label": "modified-timestamp", "root": "attribute" } },
      { "value": "2023-10-10T13:22:37.000", "value_type": "datetime", "type": { "label": "modified-timestamp", "root": "attribute" } },
      { "value": "2023-10-10T13:39:19.000", "value_type": "datetime", "type": { "label": "modified-timestamp", "root": "attribute" } },
      { "value": "2023-11-29T09:17:47.000", "value_type": "datetime", "type": { "label": "modified-timestamp", "root": "attribute" } },
      { "value": "2023-06-14T22:44:22.000", "value_type": "datetime", "type": { "label": "created-timestamp", "root": "attribute" } }
    ],
    "id": [ { "value": "/vaticle/research/prototypes/nlp-query-generator.py", "value_type": "string", "type": { "label": "path", "root": "attribute" } } ],
    "type": { "label": "file", "root": "entity" }
  }
}
{
  "resource": {
    "event-timestamp": [
      { "value": "2023-11-23T07:05:18.000", "value_type": "datetime", "type": { "label": "modified-timestamp", "root": "attribute" } },
      { "value": "2023-10-01T01:46:23.000", "value_type": "datetime", "type": { "label": "created-timestamp", "root": "attribute" } }
    ],
    "id": [ { "value": "/vaticle/engineering/projects/typedb-3.0", "value_type": "string", "type": { "label": "path", "root": "attribute" } } ],
    "type": { "label": "directory", "root": "entity" }
  }
}
```


Querying with inheritance polymorphism

"List the ID and event timestamps of every resource."

```
{
  "resource": {
    "event-timestamp": [
      { "value": "2023-08-08T11:57:54.000", "value_type": "datetime", "type": { "label": "modified-timestamp", "root": "attribute" } },
      { "value": "2023-08-13T13:16:10.000", "value_type": "datetime", "type": { "label": "modified-timestamp", "root": "attribute" } },
      { "value": "2023-10-10T13:22:37.000", "value_type": "datetime", "type": { "label": "modified-timestamp", "root": "attribute" } },
      { "value": "2023-10-10T13:39:19.000", "value_type": "datetime", "type": { "label": "modified-timestamp", "root": "attribute" } },
      { "value": "2023-11-29T09:17:47.000", "value_type": "datetime", "type": { "label": "modified-timestamp", "root": "attribute" } },
      { "value": "2023-06-14T22:44:22.000", "value_type": "datetime", "type": { "label": "created-timestamp", "root": "attribute" } }
    ],
    "id": [ { "value": "/vaticle/research/prototypes/nlp-query-generator.py", "value_type": "string", "type": { "label": "path", "root": "attribute" } } ],
    "type": { "label": "file", "root": "entity" }
  }
}
```

file is a subtype of resource

```
{
  "resource": {
    "event-timestamp": [
      { "value": "2023-11-23T07:05:18.000", "value_type": "datetime", "type": { "label": "modified-timestamp", "root": "attribute" } },
      { "value": "2023-10-01T01:46:23.000", "value_type": "datetime", "type": { "label": "created-timestamp", "root": "attribute" } }
    ],
    "id": [ { "value": "/vaticle/engineering/projects/typedb-3.0", "value_type": "string", "type": { "label": "path", "root": "attribute" } } ],
    "type": { "label": "directory", "root": "entity" }
  }
}
```

directory is a subtype of resource

Querying with inheritance polymorphism

"List the ID and event timestamps of every resource."

```
{
  "resource": {
    "event-timestamp": [
      { "value": "2023-08-08T11:57:54.000", "value_type": "datetime", "type": { "label": "modified-timestamp", "root": "attribute" } },
      { "value": "2023-08-13T13:16:10.000", "value_type": "datetime", "type": { "label": "modified-timestamp", "root": "attribute" } },
      { "value": "2023-10-10T13:22:37.000", "value_type": "datetime", "type": { "label": "modified-timestamp", "root": "attribute" } },
      { "value": "2023-10-10T13:39:19.000", "value_type": "datetime", "type": { "label": "modified-timestamp", "root": "attribute" } },
      { "value": "2023-11-29T09:17:47.000", "value_type": "datetime", "type": { "label": "modified-timestamp", "root": "attribute" } },
      { "value": "2023-06-14T22:44:22.000", "value_type": "datetime", "type": { "label": "created-timestamp", "root": "attribute" } }
    ],
    "id": [ { "value": "/vaticle/research/prototypes/nlp-query-generator.py", "value_type": "string", "type": { "label": "path", "root": "attribute" } } ],
    "type": { "label": "file", "root": "entity" }
  }
}
{
  "resource": {
    "event-timestamp": [
      { "value": "2023-11-23T07:05:18.000", "value_type": "datetime", "type": { "label": "modified-timestamp", "root": "attribute" } },
      { "value": "2023-10-01T01:46:23.000", "value_type": "datetime", "type": { "label": "created-timestamp", "root": "attribute" } }
    ],
    "id": [ { "value": "/vaticle/engineering/projects/typedb-3.0", "value_type": "string", "type": { "label": "path", "root": "attribute" } } ],
    "type": { "label": "directory", "root": "entity" }
  }
}
```

path is a subtype of id



Querying with inheritance polymorphism

"List the ID and event timestamps of every resource."

modified-timestamp is a
subtype of event-timestamp

```
{
  "resource": {
    "event-timestamp": [
      { "value": "2023-08-08T11:57:54.000", "value_type": "datetime", "type": { "label": "modified-timestamp", "root": "attribute" } },
      { "value": "2023-08-13T13:16:10.000", "value_type": "datetime", "type": { "label": "modified-timestamp", "root": "attribute" } },
      { "value": "2023-10-10T13:22:37.000", "value_type": "datetime", "type": { "label": "modified-timestamp", "root": "attribute" } },
      { "value": "2023-10-10T13:39:19.000", "value_type": "datetime", "type": { "label": "modified-timestamp", "root": "attribute" } },
      { "value": "2023-11-29T09:17:47.000", "value_type": "datetime", "type": { "label": "modified-timestamp", "root": "attribute" } },
      { "value": "2023-06-14T22:44:22.000", "value_type": "datetime", "type": { "label": "created-timestamp", "root": "attribute" } }
    ],
    "id": [ { "value": "/vaticle/research/prototypes/nlp-query-generator.py", "value_type": "string", "type": { "label": "path", "root": "attribute" } } ],
    "type": { "label": "file", "root": "entity" }
  }
}
{
  "resource": {
    "event-timestamp": [
      { "value": "2023-11-23T07:05:18.000", "value_type": "datetime", "type": { "label": "modified-timestamp", "root": "attribute" } },
      { "value": "2023-10-01T01:46:23.000", "value_type": "datetime", "type": { "label": "created-timestamp", "root": "attribute" } }
    ],
    "id": [ { "value": "/vaticle/engineering/projects/typedb-3.0", "value_type": "string", "type": { "label": "path", "root": "attribute" } } ],
    "type": { "label": "directory", "root": "entity" }
  }
}
```

created-timestamp is a subtype of event-timestamp

Querying with **interface** polymorphism

"List the ID and created timestamps of everything that was created."

```
match  
$x has created-timestamp $created;  
fetch  
$x: id, created-timestamp;
```

The type of `$x` is not explicitly specified

Querying with **interface** polymorphism

"List the ID and created timestamps of everything that was created."

```
{
  "x": {
    "created-timestamp": [ { "value": "2023-01-31T19:39:32.000", "value_type": "datetime", "type": { "label": "created-timestamp", "root": "attribute" } } ],
    "id": [ { "value": "/vaticle/research/prototypes/root-cause-analyzer.py", "value_type": "string", "type": { "label": "path", "root": "attribute" } } ],
    "type": { "label": "file", "root": "entity" }
  }
}
{
  "x": {
    "created-timestamp": [ { "value": "2023-03-11T02:30:39.000", "value_type": "datetime", "type": { "label": "created-timestamp", "root": "attribute" } } ],
    "id": [ { "value": "reginald@vaticle.com", "value_type": "string", "type": { "label": "email", "root": "attribute" } } ],
    "type": { "label": "user", "root": "entity" }
  }
}
{
  "x": {
    "created-timestamp": [ { "value": "2023-09-07T17:47:57.000", "value_type": "datetime", "type": { "label": "created-timestamp", "root": "attribute" } } ],
    "id": [ { "value": "engineers", "value_type": "string", "type": { "label": "name", "root": "attribute" } } ],
    "type": { "label": "user-group", "root": "entity" }
  }
}
```

Querying with **interface** polymorphism

"List the ID and created timestamps of everything that was created."

```
{
  "x": {
    "created-timestamp": [ { "value": "2023-01-31T19:39:32.000", "value_type": "datetime", "type": { "label": "created-timestamp", "root": "attribute" } } ],
    "id": [ { "value": "/vaticle/research/prototypes/root-cause-analyzer.py", "value_type": "string", "type": { "label": "path", "root": "attribute" } } ],
    "type": { "label": "file", "root": "entity" }
  }
}
{
  "x": {
    "created-timestamp": [ { "value": "2023-03-11T02:30:39.000", "value_type": "datetime", "type": { "label": "created-timestamp", "root": "attribute" } } ],
    "id": [ { "value": "reginald@vaticle.com", "value_type": "string", "type": { "label": "email", "root": "attribute" } } ],
    "type": { "label": "user", "root": "entity" }
  }
}
{
  "x": {
    "created-timestamp": [ { "value": "2023-09-07T17:47:57.000", "value_type": "datetime", "type": { "label": "created-timestamp", "root": "attribute" } } ],
    "id": [ { "value": "engineers", "value_type": "string", "type": { "label": "name", "root": "attribute" } } ],
    "type": { "label": "user-group", "root": "entity" }
  }
}
```


file, user, and user-group all own created-timestamp

Querying with **interface** polymorphism

"List the ID and owner ID of every resource."

```
match  
(resource: $resource, resource-owner: $owner) isa resource-ownership;  
fetch  
$resource: id;  
$owner: id;
```

The types of `$resource` and `$owner` are not explicitly specified



Querying with **interface** polymorphism

"List the ID and owner ID of every resource."

```
{
  "owner": {
    "id": [ { "value": "engineers", "value_type": "string", "type": { "label": "name", "root": "attribute" } } ],
    "type": { "label": "user-group", "root": "entity" }
  },
  "resource": {
    "id": [ { "value": "/vaticle/engineering", "value_type": "string", "type": { "label": "path", "root": "attribute" } } ],
    "type": { "label": "directory", "root": "entity" }
  }
}
{
  "owner": {
    "id": [ { "value": "cedric@vaticle.com", "value_type": "string", "type": { "label": "email", "root": "attribute" } } ],
    "type": { "label": "admin", "root": "entity" }
  },
  "resource": {
    "id": [ { "value": "/vaticle/engineering/projects/typedb-3.0/traversal-engine.rs", "value_type": "string", "type": { "label": "path", "root": "attribute" } } ],
    "type": { "label": "file", "root": "entity" }
  }
}
```


Querying with **interface** polymorphism

"List the ID and owner ID of every resource."

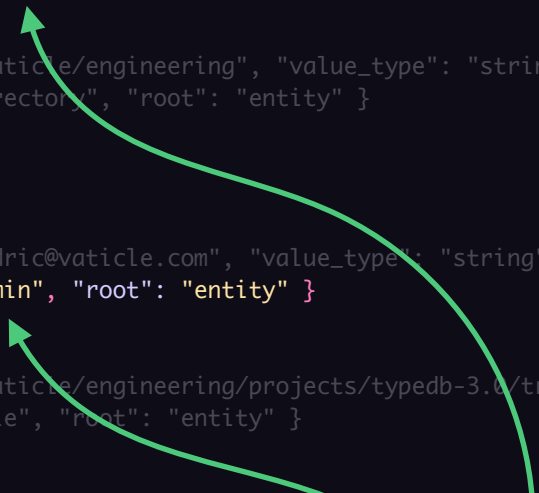
```
{
  "owner": {
    "id": [ { "value": "engineers", "value_type": "string", "type": { "label": "name", "root": "attribute" } } ],
    "type": { "label": "user-group", "root": "entity" }
  },
  "resource": {
    "id": [ { "value": "/vaticle/engineering", "value_type": "string", "type": { "label": "path", "root": "attribute" } } ],
    "type": { "label": "directory", "root": "entity" }
  }
}
{
  "owner": {
    "id": [ { "value": "cedric@vaticle.com", "value_type": "string", "type": { "label": "email", "root": "attribute" } } ],
    "type": { "label": "admin", "root": "entity" }
  },
  "resource": {
    "id": [ { "value": "/vaticle/engineering/projects/typech-3.0/traversal-engine.rs", "value_type": "string", "type": { "label": "path", "root": "attribute" } } ],
    "type": { "label": "file", "root": "entity" }
  }
}
```

directory and file both play resource in resource-ownership

Querying with **interface** polymorphism

"List the ID and owner ID of every resource."

```
{
  "owner": {
    "id": [ { "value": "engineers", "value_type": "string", "type": { "label": "name", "root": "attribute" } } ],
    "type": { "label": "user-group", "root": "entity" }
  },
  "resource": {
    "id": [ { "value": "/vaticle/engineering", "value_type": "string", "type": { "label": "path", "root": "attribute" } } ],
    "type": { "label": "directory", "root": "entity" }
  }
}
{
  "owner": {
    "id": [ { "value": "cedric@vaticle.com", "value_type": "string", "type": { "label": "email", "root": "attribute" } } ],
    "type": { "label": "admin", "root": "entity" }
  },
  "resource": {
    "id": [ { "value": "/vaticle/engineering/projects/typedb-3.0/traversal-engine.rs", "value_type": "string", "type": { "label": "path", "root": "attribute" } } ],
    "type": { "label": "file", "root": "entity" }
  }
}
```



user-group and admin both play resource-owner in resource-ownership

Querying with **interface** polymorphism

"List the ID and owner ID of every resource."

```
match  
(resource: $resource, resource-owner: $owner) isa resource-ownership;  
fetch  
$resource: id;  
$owner: id;
```

Possible pairs of return types

\$resource	\$owner
file	user
file	admin
file	user-group
directory	user
directory	admin
directory	user-group

Querying with **parametric** polymorphism

"List the type and ID of everything that has a multivalued attribute."

```
match
  $x has $attribute-1;
  $x has $attribute-2;
  $attribute-1 isa! $attribute-type;
  $attribute-2 isa! $attribute-type;
  not { $attribute-1 is $attribute-2; };
  fetch
  $attribute-type;
"x-type": { match $x isa! $t; fetch $t; };
"x-id": { match $x has id $id; fetch $id; };
```

The type of `$attribute-1` and `$attribute-2` has been *variablized*

The `fetch` clause only explicitly describes the `id` type

The `match` clause contains no explicit types
(completely independent of schema)

Querying with **parametric** polymorphism

"List the type and ID of everything that has a multivalued attribute."

```
{
  "attribute-type": { "label": "modified-timestamp", "root": "attribute" },
  "x-id": [ { "id": { "value": "/vaticle/engineering/tools/data-generator.rs", "value_type": "string", "type": { "label": "path", "root": "attribute" } } } ],
  "x-type": [ { "t": { "label": "file", "root": "entity" } } ]
}
{
  "attribute-type": { "label": "modified-timestamp", "root": "attribute" },
  "x-id": [ { "id": { "value": "/vaticle/research/papers", "value_type": "string", "type": { "label": "path", "root": "attribute" } } } ],
  "x-type": [ { "t": { "label": "directory", "root": "entity" } } ]
}
{
  "attribute-type": { "label": "modified-timestamp", "root": "attribute" },
  "x-id": [ { "id": { "value": "/vaticle/engineering/tools", "value_type": "string", "type": { "label": "path", "root": "attribute" } } } ],
  "x-type": [ { "t": { "label": "directory", "root": "entity" } } ]
}
```

Querying with **parametric** polymorphism

"List the type and ID of everything that has a multivalued attribute."

```
{
  "attribute-type": { "label": "modified-timestamp", "root": "attribute" },
  "x-id": [ { "id": { "value": "/vaticle/engineering/tools/data-generator.rs", "value_type": "string", "type": { "label": "path", "root": "attribute" } } } ],
  "x-type": [ { "t": { "label": "file", "root": "entity" } } ]
}
{
  "attribute-type": { "label": "modified-timestamp", "root": "attribute" },
  "x-id": [ { "id": { "value": "/vaticle/research/papers", "value_type": "string", "type": { "label": "path", "root": "attribute" } } } ],
  "x-type": [ { "t": { "label": "directory", "root": "entity" } } ]
}
{
  "attribute-type": { "label": "modified-timestamp", "root": "attribute" },
  "x-id": [ { "id": { "value": "/vaticle/engineering/tools", "value_type": "string", "type": { "label": "path", "root": "attribute" } } } ],
  "x-type": [ { "t": { "label": "directory", "root": "entity" } } ]
}
```

modified-timestamp is the only multivalued attribute at this point

Querying the schema

"List the attribute types that files can have."

```
match  
file owns $attribute-type;  
fetch  
$attribute-type;
```

Types are queried using schema keywords like "owns"

Querying the schema

"List the attribute types that files can have."

```
{ "attribute-type": { "label": "modified-timestamp", "root": "attribute" } }  
{ "attribute-type": { "label": "path", "root": "attribute" } }  
{ "attribute-type": { "label": "created-timestamp", "root": "attribute" } }
```


Querying the schema

"List the concrete types that can play roles in ownerships."

```
match
ownership-type sub ownership;
ownership-type relates $role;
$player plays $role;
not { $player abstract; };
fetch
$role;
$player;
```

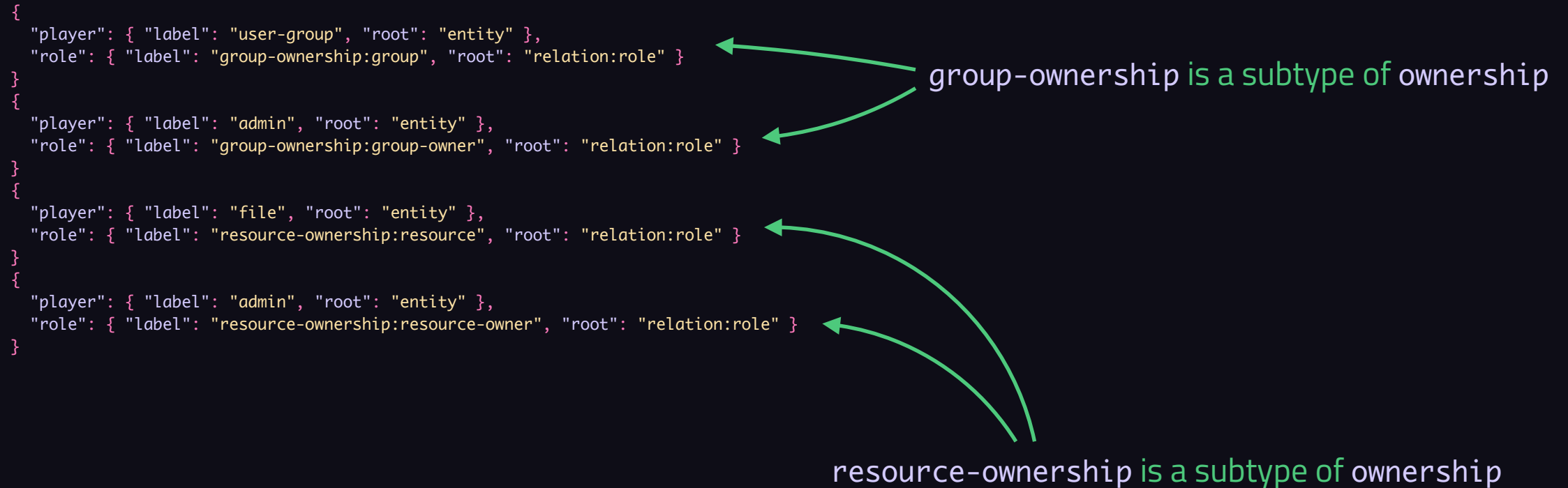
Type hierarchies are queried with "sub"

Relation roles are queried with "relates"

Roleplayers are queried with "plays"

Querying the schema

"List the concrete types that can play roles in ownerships."



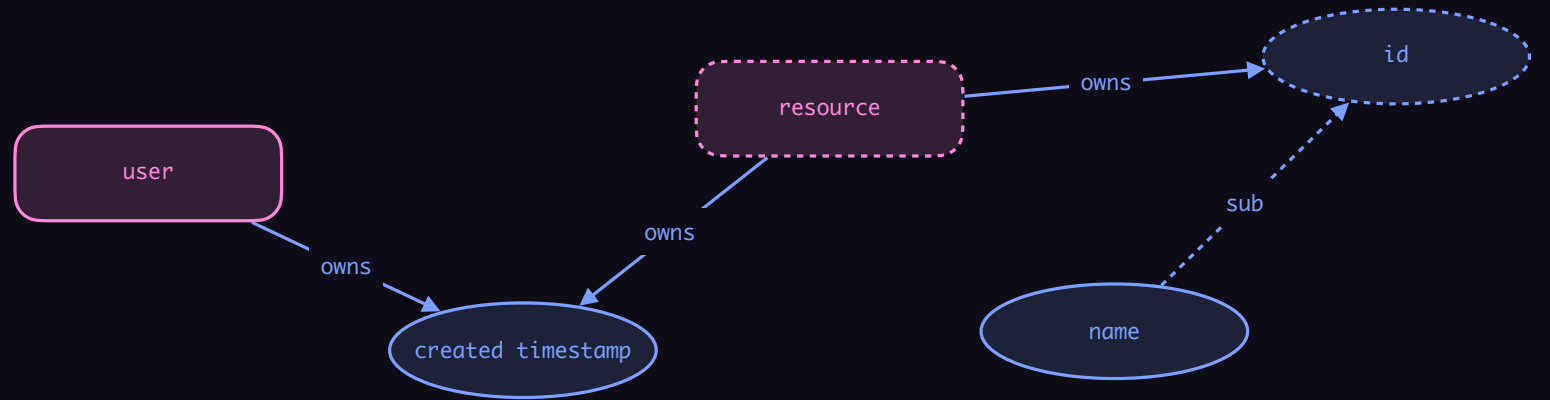
Extending the schema

```
define
repository sub resource,
  owns name as id,
  plays commit:repository;

commit sub relation,
  relates repository,
  relates author,
  owns hash,
  owns created-timestamp;

user plays commit:author;

hash sub id;
```



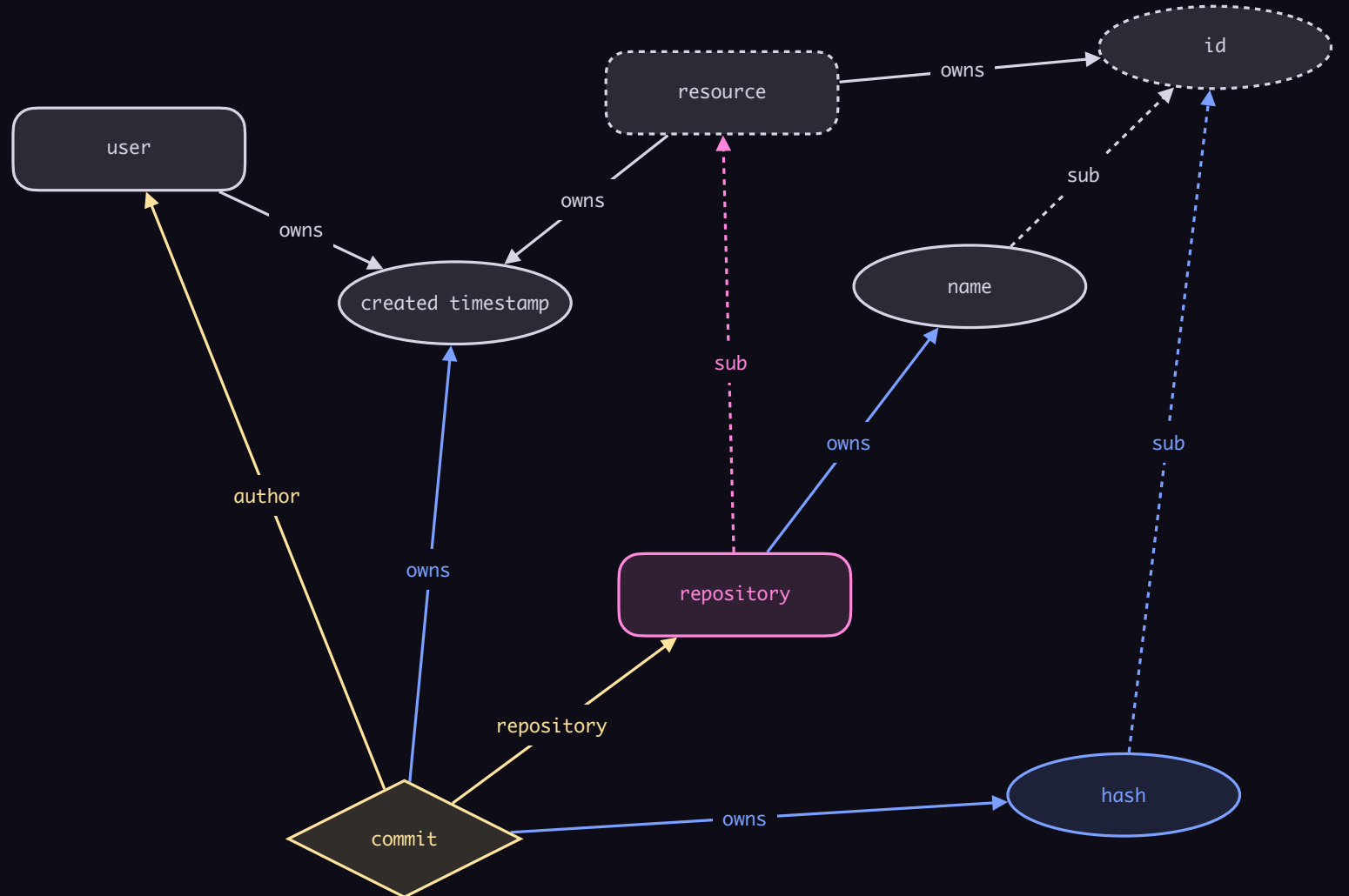
Extending the schema

```
define
repository sub resource,
  owns name as id,
  plays commit:repository;

commit sub relation,
  relates repository,
  relates author,
  owns hash,
  owns created-timestamp;

user plays commit:author;

hash sub id;
```



Extending the schema

```

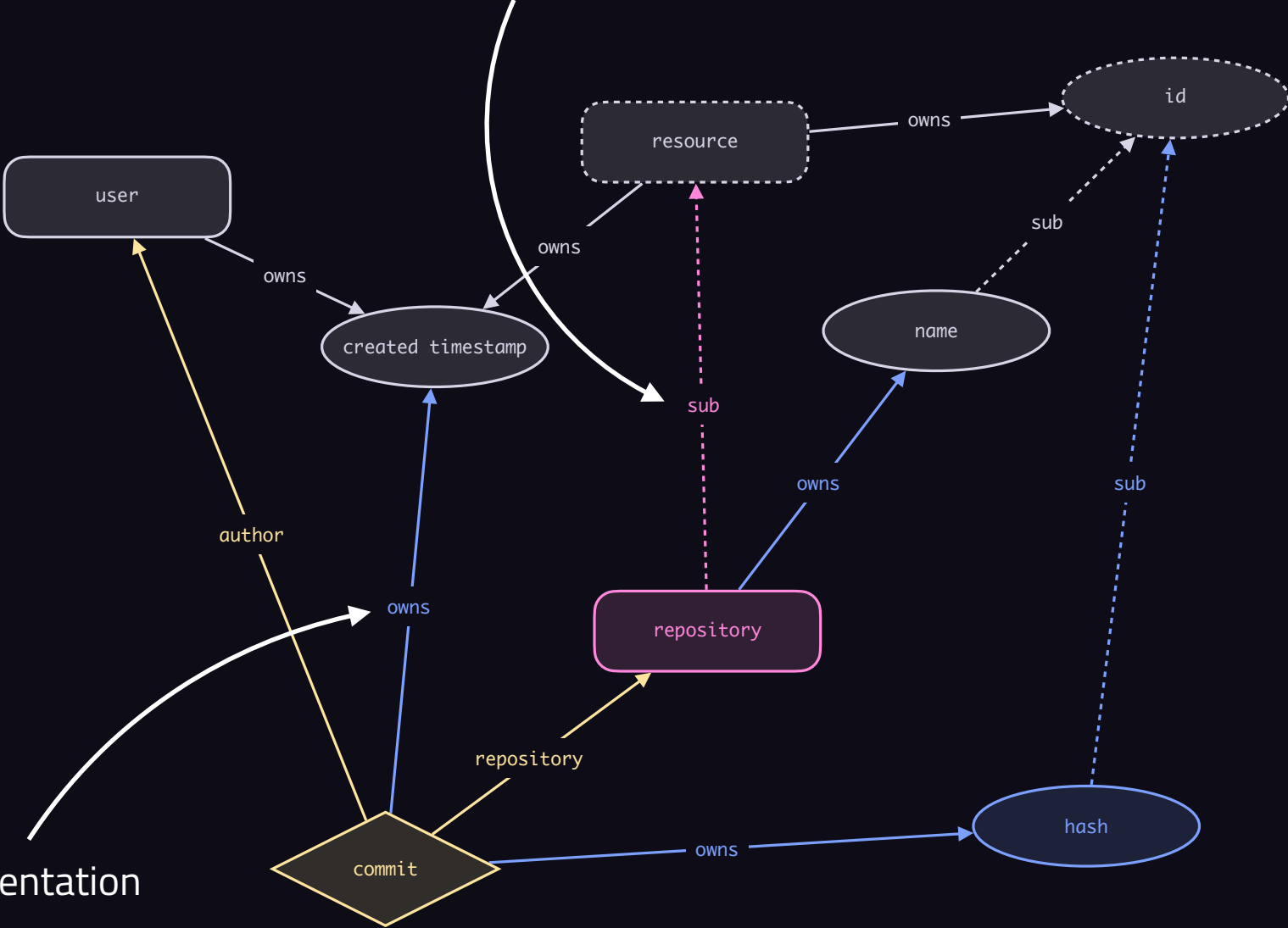
define
repository sub resource,
  owns name as id,
  plays commit:repository;

commit sub relation,
  relates repository,
  relates author,
  owns hash,
  owns created-timestamp;

user plays commit:author;

hash sub id;
  
```

resource has a new subtype



created-timestamp has a new implementation

Extending the schema

"List the ID and event timestamps of every resource."

```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```

Extending the schema

"List the ID and event timestamps of every resource."

```
{
  "resource": {
    "event-timestamp": [ { "value": "2023-05-20T19:12:11.000", "value_type": "datetime", "type": { "label": "created-timestamp", "root": "attribute" } } ],
    "id": [ { "value": "/vaticle/engineering/projects/typedb-3.0/query-parser.rs", "value_type": "string", "type": { "label": "path", "root": "attribute" } } ],
    "type": { "label": "file", "root": "entity" }
  }
}
{
  "resource": {
    "event-timestamp": [
      { "value": "2023-11-23T07:05:18.000", "value_type": "datetime", "type": { "label": "modified-timestamp", "root": "attribute" } },
      { "value": "2023-10-01T01:46:23.000", "value_type": "datetime", "type": { "label": "created-timestamp", "root": "attribute" } }
    ],
    "id": [ { "value": "/vaticle/engineering/projects/typedb-3.0", "value_type": "string", "type": { "label": "path", "root": "attribute" } } ],
    "type": { "label": "directory", "root": "entity" }
  }
}
{
  "resource": {
    "event-timestamp": [ { "value": "2023-06-05T14:13:23.000", "value_type": "datetime", "type": { "label": "created-timestamp", "root": "attribute" } } ],
    "id": [ { "value": "typedb-cloud", "value_type": "string", "type": { "label": "name", "root": "attribute" } } ],
    "type": { "label": "repository", "root": "entity" }
  }
}
```

Extending the schema

"List the ID and event timestamps of every resource."

```
{
  "resource": {
    "event-timestamp": [ { "value": "2023-05-20T19:12:11.000", "value_type": "datetime", "type": { "label": "created-timestamp", "root": "attribute" } } ],
    "id": [ { "value": "/vaticle/engineering/projects/typedb-3.0/query-parser.rs", "value_type": "string", "type": { "label": "path", "root": "attribute" } } ],
    "type": { "label": "file", "root": "entity" }
  }
}
{
  "resource": {
    "event-timestamp": [
      { "value": "2023-11-23T07:05:18.000", "value_type": "datetime", "type": { "label": "modified-timestamp", "root": "attribute" } },
      { "value": "2023-10-01T01:46:23.000", "value_type": "datetime", "type": { "label": "created-timestamp", "root": "attribute" } }
    ],
    "id": [ { "value": "/vaticle/engineering/projects/typedb-3.0", "value_type": "string", "type": { "label": "path", "root": "attribute" } } ],
    "type": { "label": "directory", "root": "entity" }
  }
}
{
  "resource": {
    "event-timestamp": [ { "value": "2023-06-05T14:13:23.000", "value_type": "datetime", "type": { "label": "created-timestamp", "root": "attribute" } } ],
    "id": [ { "value": "typedb-cloud", "value_type": "string", "type": { "label": "name", "root": "attribute" } } ],
    "type": { "label": "repository", "root": "entity" }
  }
}
```

Repositories are now returned as well!

Extending the schema

"List the ID and created timestamps of everything that was created."

```
match
$x has created-timestamp $created;
fetch
$x: id, created-timestamp;
```

Extending the schema

"List the ID and created timestamps of everything that was created."

```
{
  "x": {
    "created-timestamp": [ { "value": "2023-06-05T14:13:23.000", "value_type": "datetime", "type": { "label": "created-timestamp", "root": "attribute" } } ],
    "id": [ { "value": "typedb-cloud", "value_type": "string", "type": { "label": "name", "root": "attribute" } } ],
    "type": { "label": "repository", "root": "entity" }
  }
}
{
  "x": {
    "created-timestamp": [ { "value": "2023-06-14T22:44:22.000", "value_type": "datetime", "type": { "label": "created-timestamp", "root": "attribute" } } ],
    "id": [ { "value": "/vaticle/research/prototypes/nlp-query-generator.py", "value_type": "string", "type": { "label": "path", "root": "attribute" } } ],
    "type": { "label": "file", "root": "entity" }
  }
}
{
  "x": {
    "created-timestamp": [ { "value": "2023-06-30T04:37:33.000", "value_type": "datetime", "type": { "label": "created-timestamp", "root": "attribute" } } ],
    "id": [ { "value": "52b1142dabd766ec2e467bc74913de41", "value_type": "string", "type": { "label": "hash", "root": "attribute" } } ],
    "type": { "label": "commit", "root": "relation" }
  }
}
```

Extending the schema

"List the ID and created timestamps of everything that was created."

```
{
  "x": {
    "created-timestamp": [ { "value": "2023-06-05T14:13:23.000", "value_type": "datetime", "type": { "label": "created-timestamp", "root": "attribute" } } ],
    "id": [ { "value": "typedb-cloud", "value_type": "string", "type": { "label": "name", "root": "attribute" } } ],
    "type": { "label": "repository", "root": "entity" }
  }
}
{
  "x": {
    "created-timestamp": [ { "value": "2023-06-14T22:44:22.000", "value_type": "datetime", "type": { "label": "created-timestamp", "root": "attribute" } } ],
    "id": [ { "value": "/vaticle/research/prototypes/nlp-query-generator.py", "value_type": "string", "type": { "label": "path", "root": "attribute" } } ],
    "type": { "label": "file", "root": "entity" }
  }
}
{
  "x": {
    "created-timestamp": [ { "value": "2023-06-30T04:37:33.000", "value_type": "datetime", "type": { "label": "created-timestamp", "root": "attribute" } } ],
    "id": [ { "value": "52b1142dabd766ec2e467bc74913de41", "value_type": "string", "type": { "label": "hash", "root": "attribute" } } ],
    "type": { "label": "commit", "root": "relation" }
  }
}
```

Repositories and commits are now returned as well!

Type-theoretic language

- Type variablization allows declarative polymorphic queries.
- Results of declarative queries extend when the schema is extended.

Strong type system

Type inference and semantic validation

Type inference

"List the ID and event timestamps of every resource."

```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```

Query returns three variables:

```
$resource
$resource: id
$resource: event-timestamp
```

Query has three return types:

```
type($resource)
type($resource: id)
type($resource: event-timestamp)
```

Note: `type()` is not actually a TypeQL function

Type inference

"List the ID and event timestamps of every resource."

```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```

`type($resource)`



Type inference

"List the ID and event timestamps of every resource."

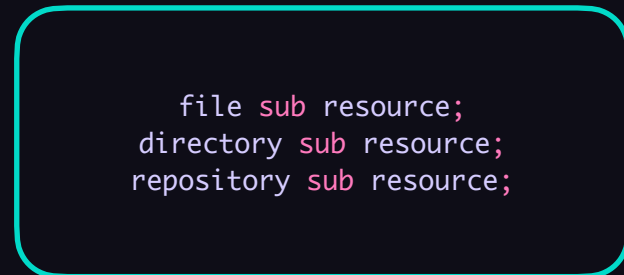
```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```



Type inference

"List the ID and event timestamps of every resource."

```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```

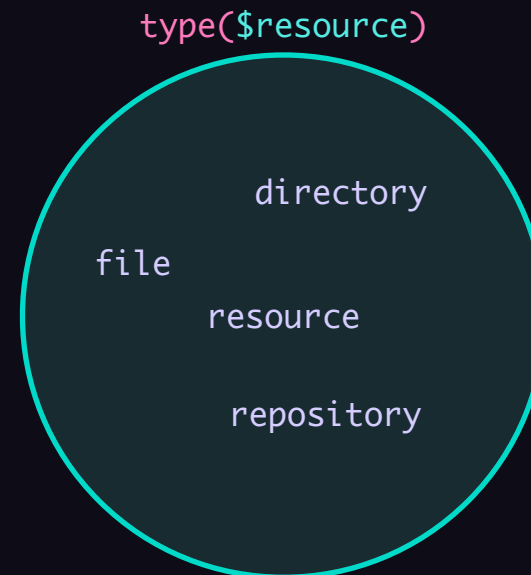


Type inference

"List the ID and event timestamps of every resource."

```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```

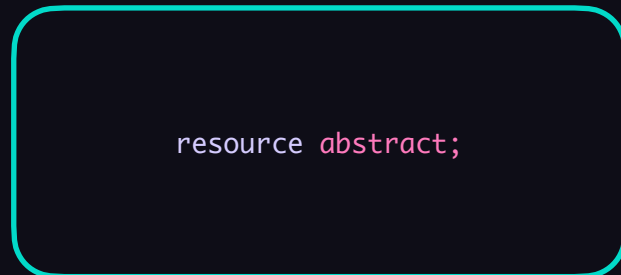
```
file sub resource;
directory sub resource;
repository sub resource;
```



Type inference

"List the ID and event timestamps of every resource."

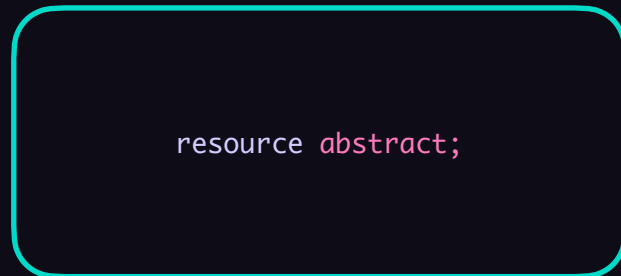
```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```



Type inference

"List the ID and event timestamps of every resource."

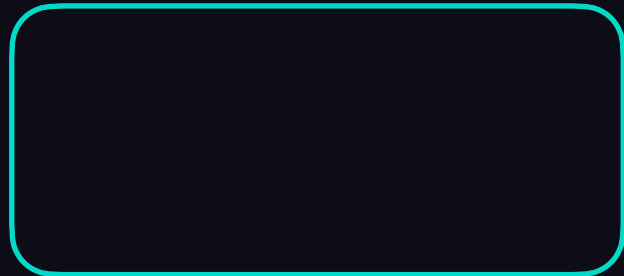
```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```



Type inference

"List the ID and event timestamps of every resource."

```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```



`type($resource)`



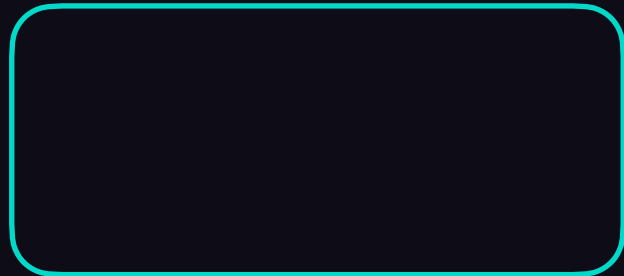
`type($resource: id) <type($resource)>`



Type inference

"List the ID and event timestamps of every resource."

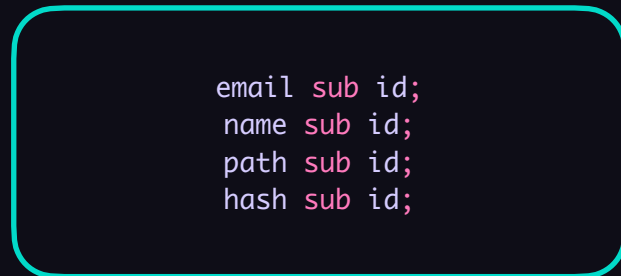
```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```



Type inference

"List the ID and event timestamps of every resource."

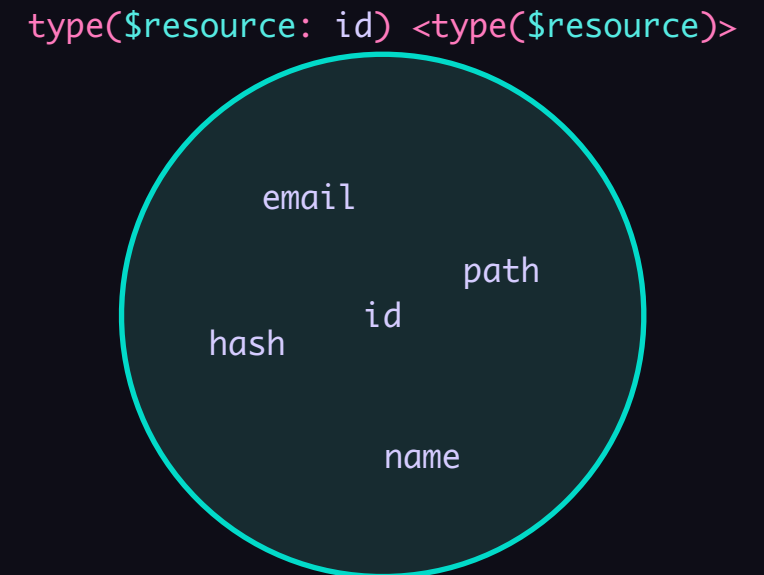
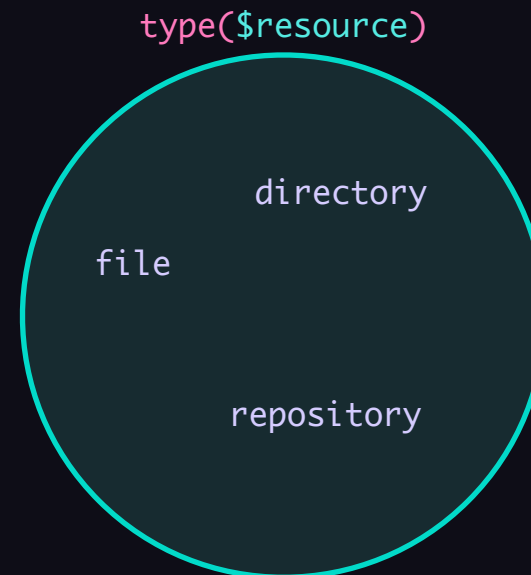
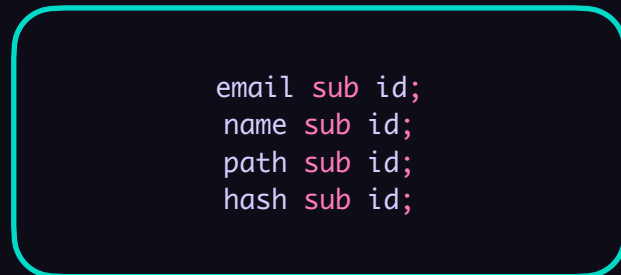
```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```



Type inference

"List the ID and event timestamps of every resource."

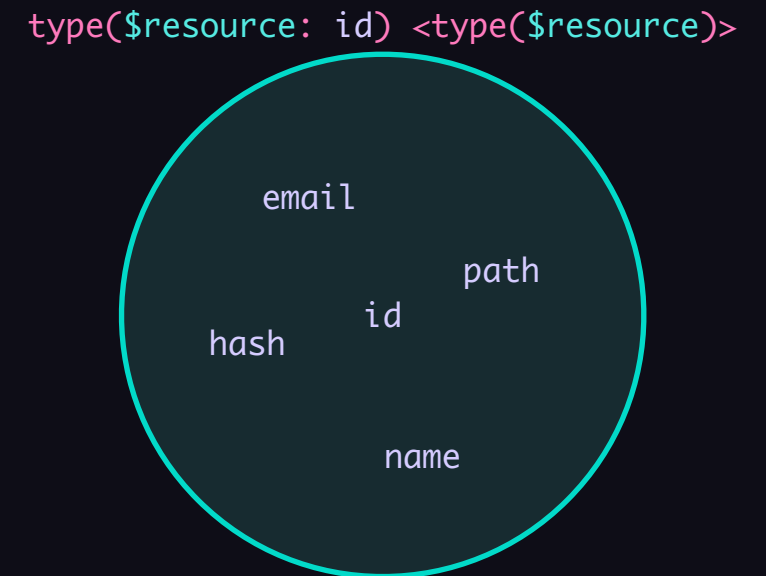
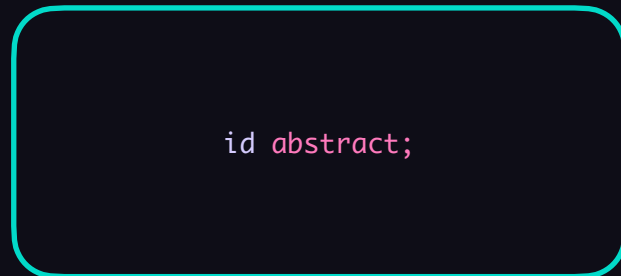
```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```



Type inference

"List the ID and event timestamps of every resource."

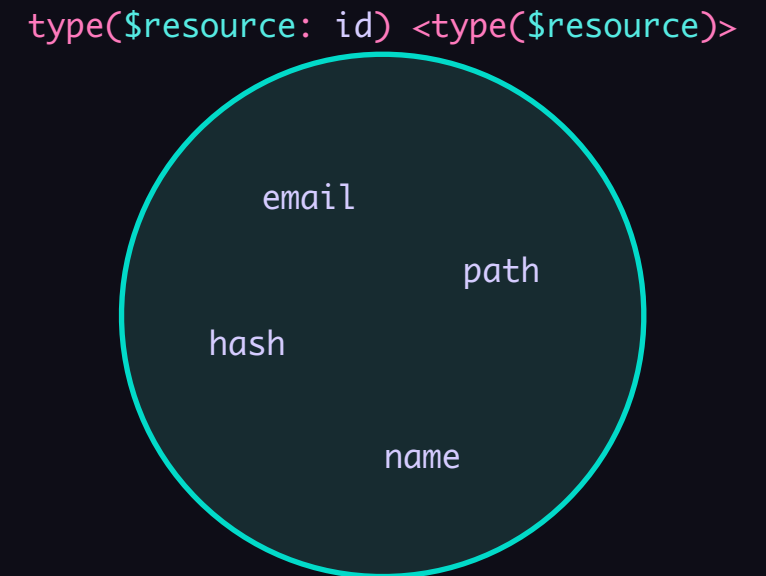
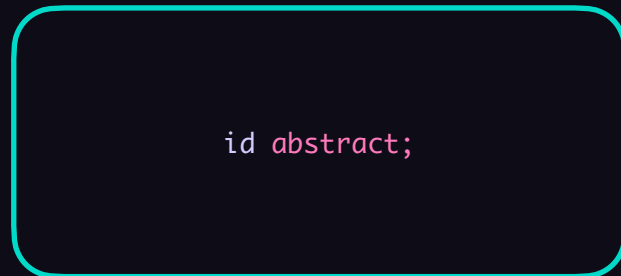
```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```



Type inference

"List the ID and event timestamps of every resource."

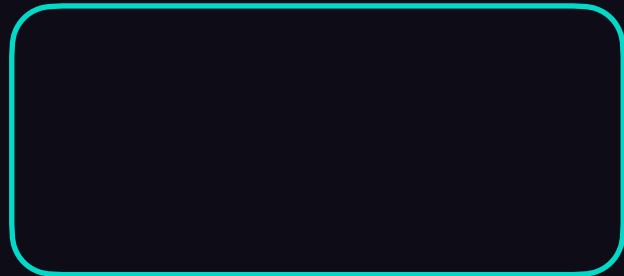
```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```



Type inference

"List the ID and event timestamps of every resource."

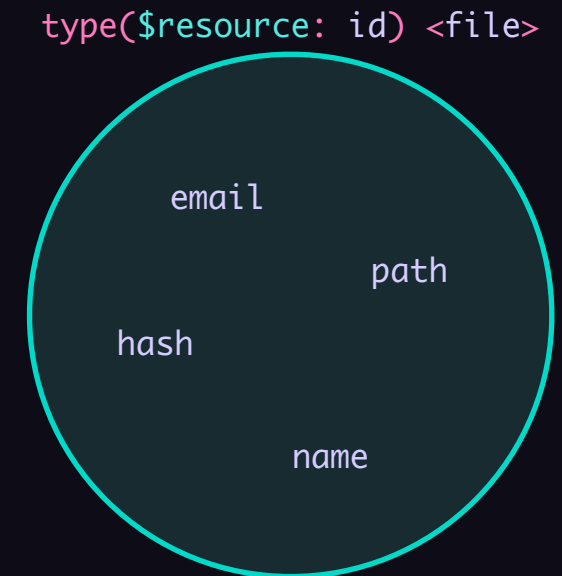
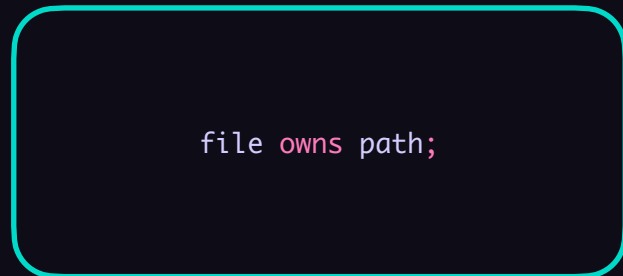
```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```



Type inference

"List the ID and event timestamps of every resource."

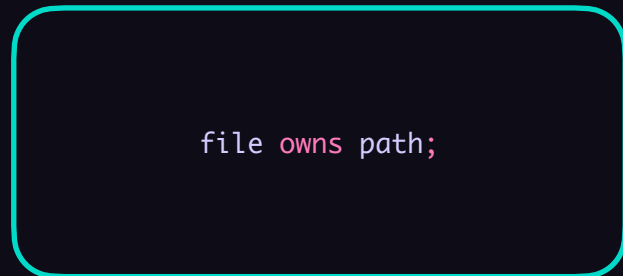
```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```



Type inference

"List the ID and event timestamps of every resource."

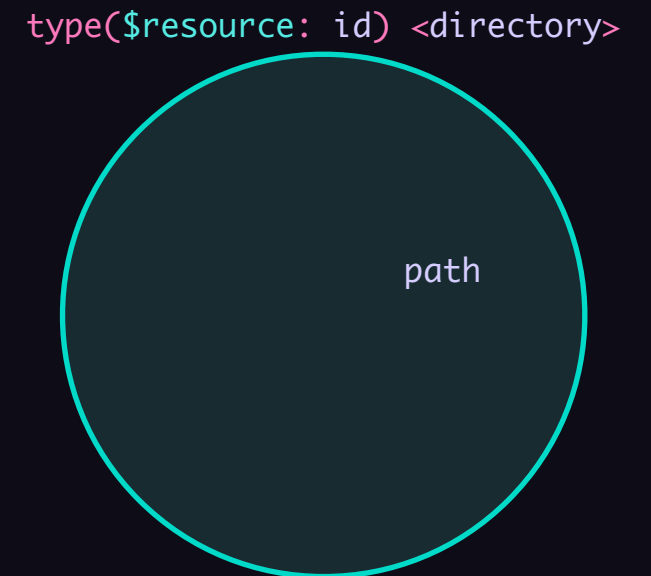
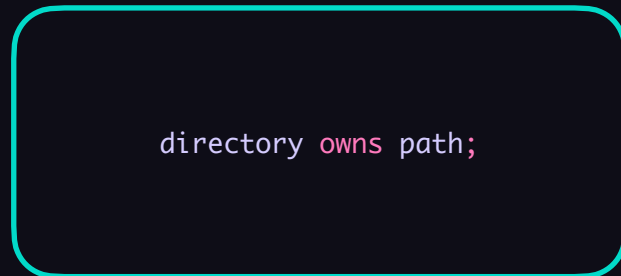
```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```



Type inference

"List the ID and event timestamps of every resource."

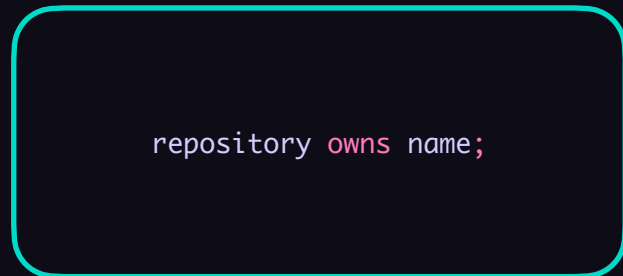
```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```



Type inference

"List the ID and event timestamps of every resource."

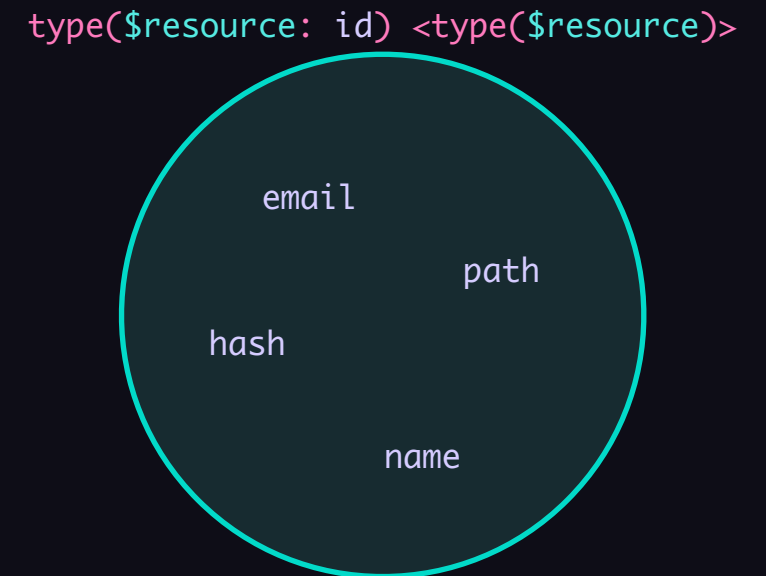
```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```



Type inference

"List the ID and event timestamps of every resource."

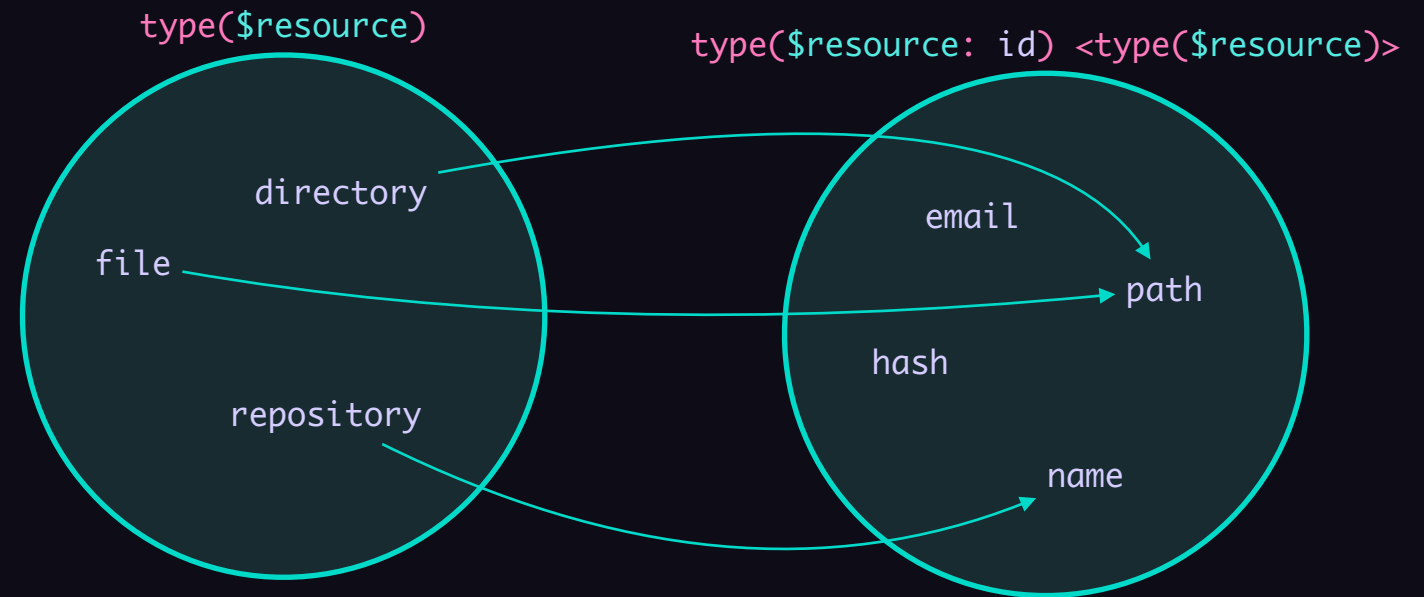
```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```



Type inference

"List the ID and event timestamps of every resource."

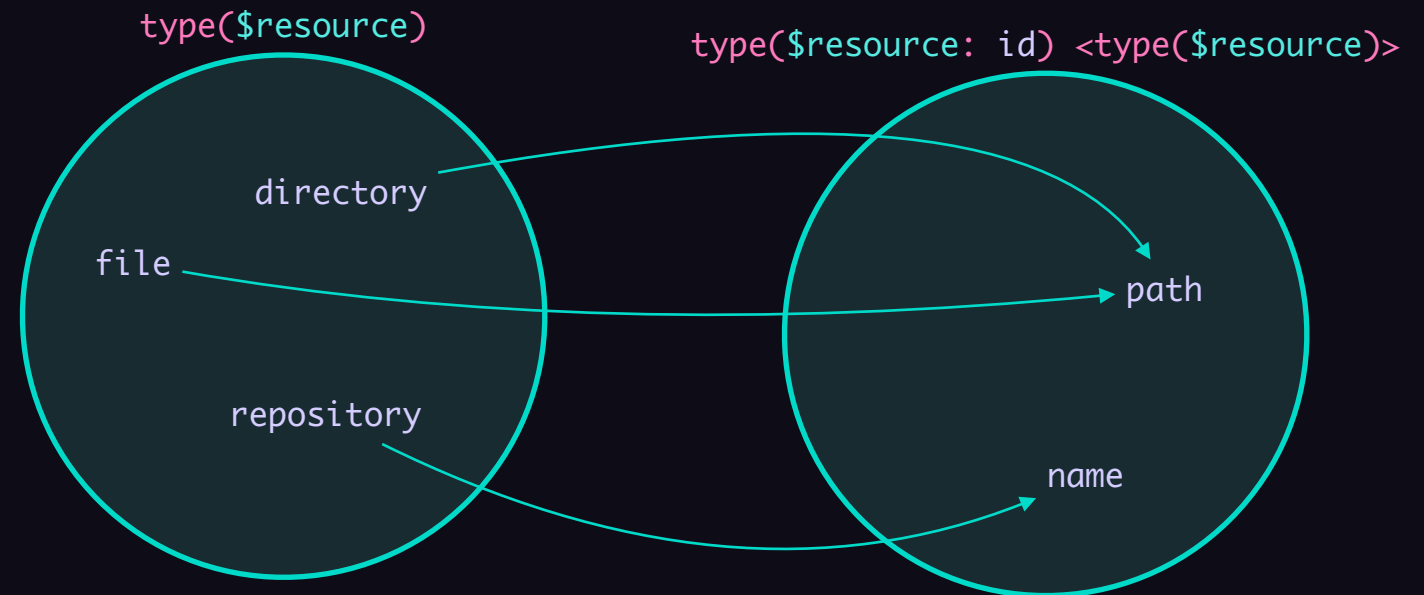
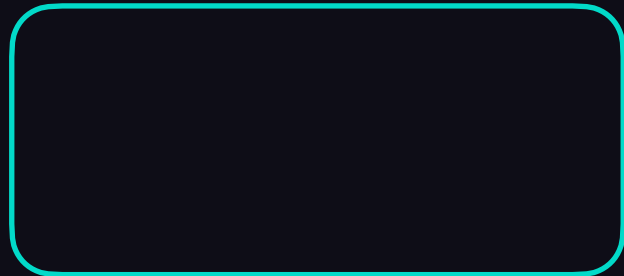
```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```



Type inference

"List the ID and event timestamps of every resource."

```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```



Type inference

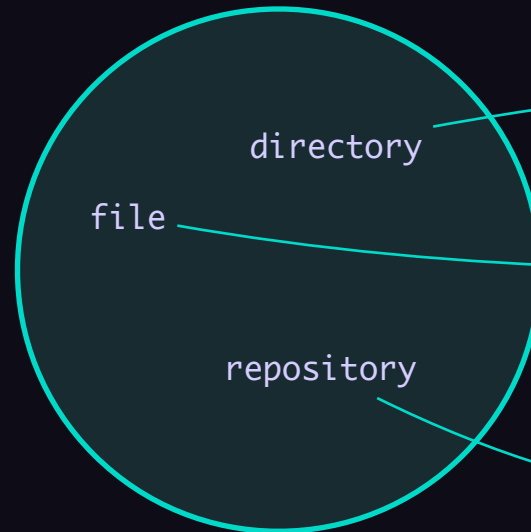
"List the ID and event timestamps of every resource."

```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```

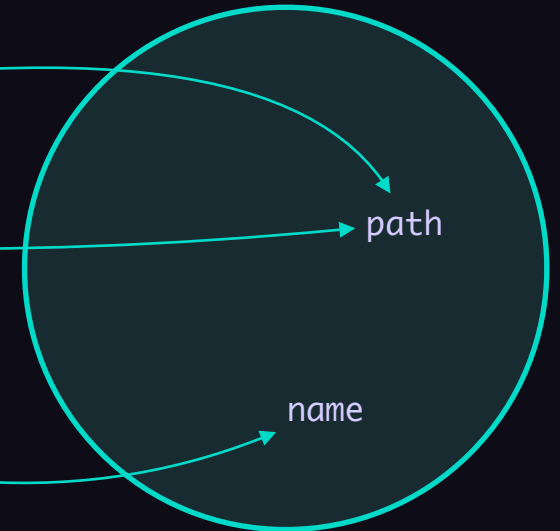
```
type($resource: event-timestamp) <type($resource)>
```



```
type($resource)
```



```
type($resource: id) <type($resource)>
```



directory

file

repository

path

name

Type inference

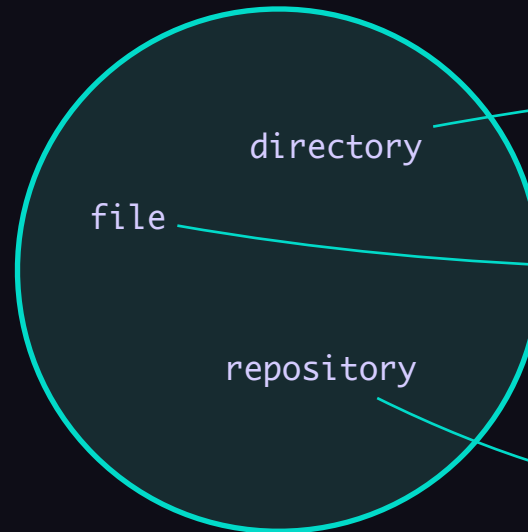
"List the ID and event timestamps of every resource."

```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```

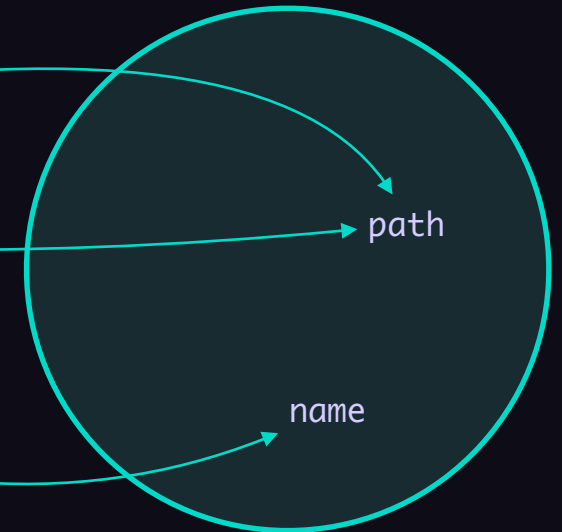
`type($resource: event-timestamp) <type($resource)>`



`type($resource)`



`type($resource: id) <type($resource)>`



Abbreviation: `type($x: y) <type($x)>` → `type($x: y)`

Type inference

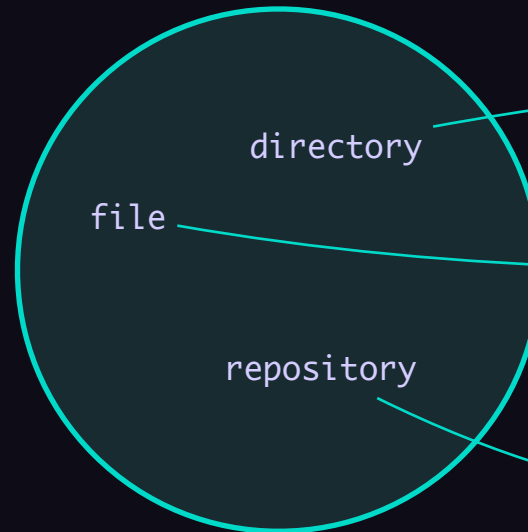
"List the ID and event timestamps of every resource."

```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```

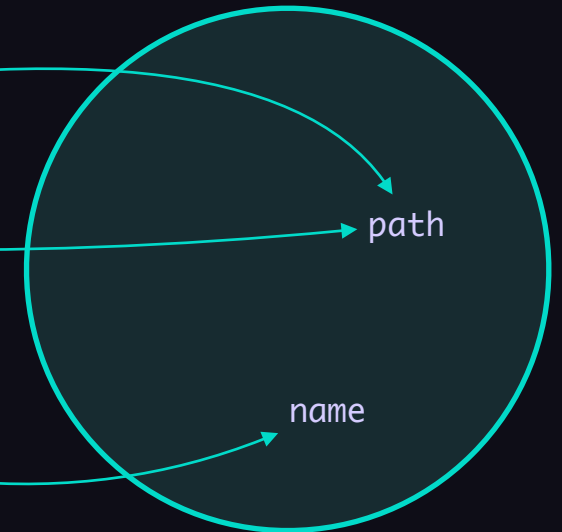
`type($resource: event-timestamp)`



`type($resource)`



`type($resource: id)`



Abbreviation: `type($x: y) <type($x)> → type($x: y)`

Type inference

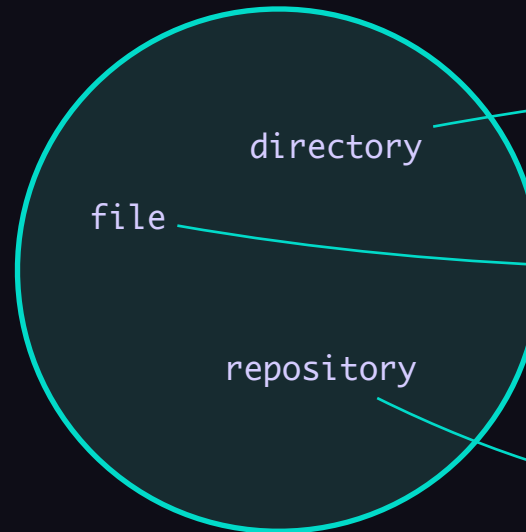
"List the ID and event timestamps of every resource."

```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```

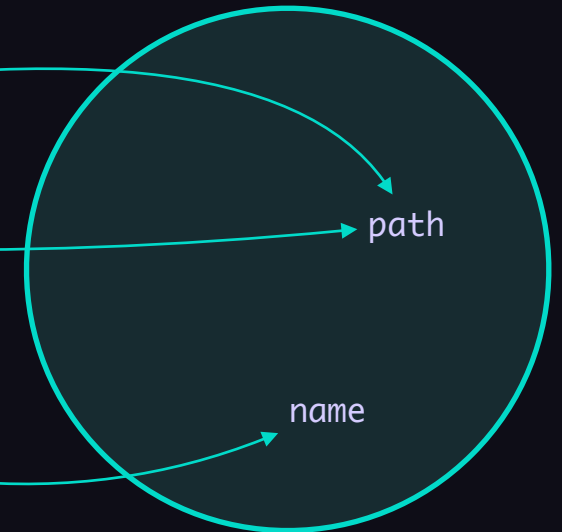
`type($resource: event-timestamp)`



`type($resource)`



`type($resource: id)`



directory

file

repository

directory

path

name

Type inference

"List the ID and event timestamps of every resource."

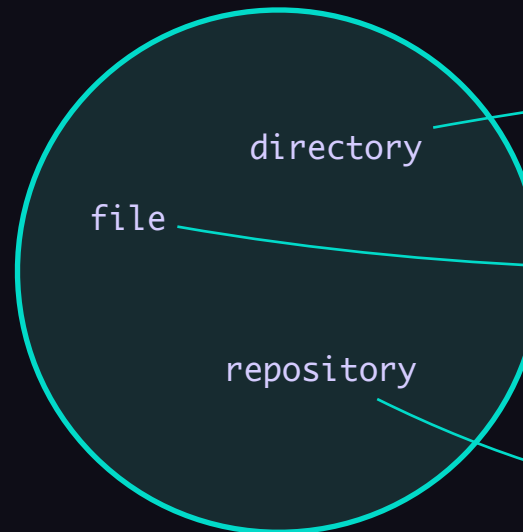
```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```

```
created-timestamp sub event-timestamp;
modified-timestamp sub event-timestamp;
login-timestamp sub event-timestamp;
```

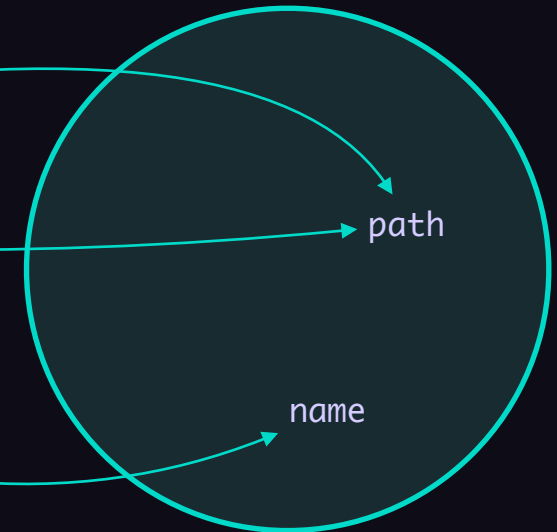
`type($resource: event-timestamp)`



`type($resource)`



`type($resource: id)`



Type inference

"List the ID and event timestamps of every resource."

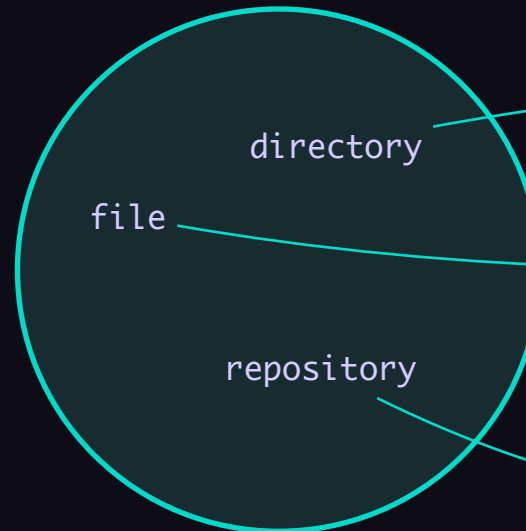
```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```

```
event-timestamp abstract;
```

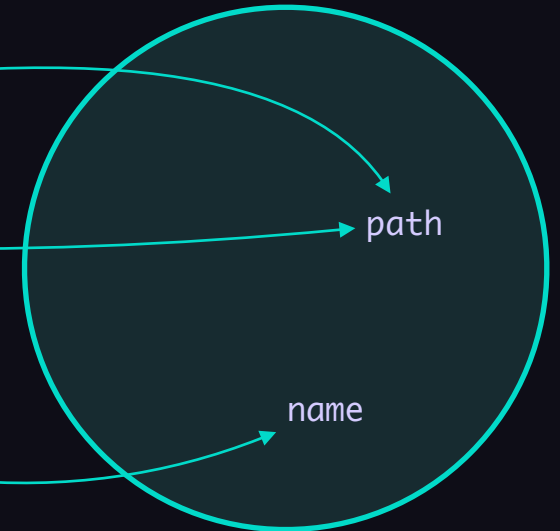
`type($resource: event-timestamp)`



`type($resource)`



`type($resource: id)`

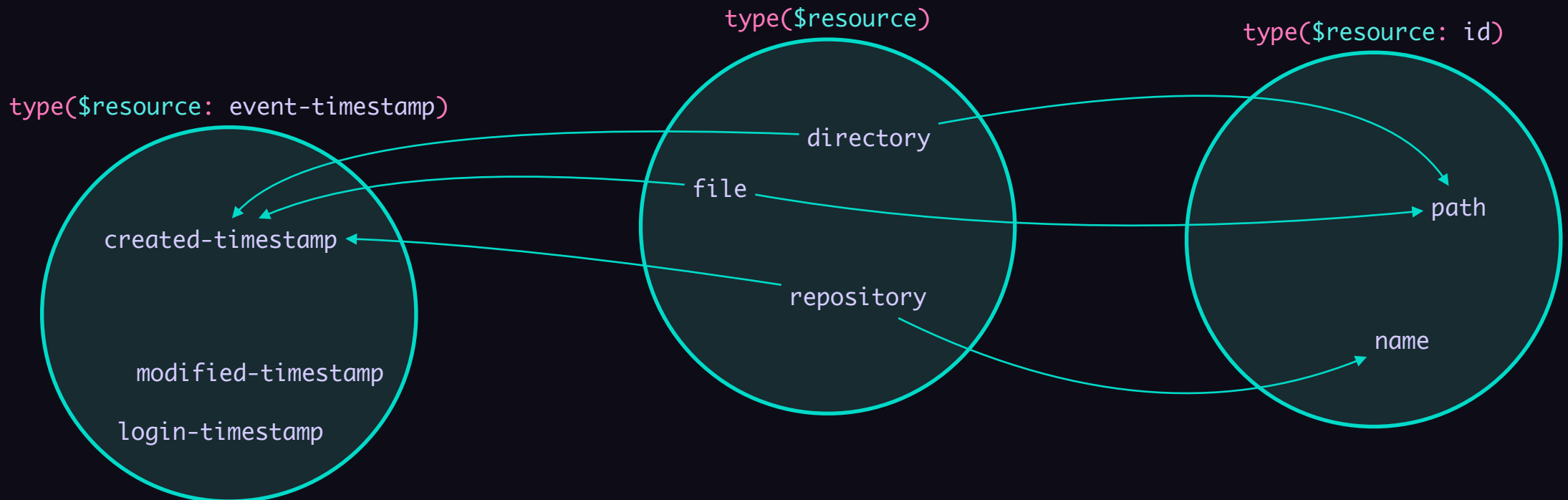


Type inference

"List the ID and event timestamps of every resource."

```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```

```
resource owns created-timestamp;
file sub resource;
directory sub resource;
repository sub resource;
```

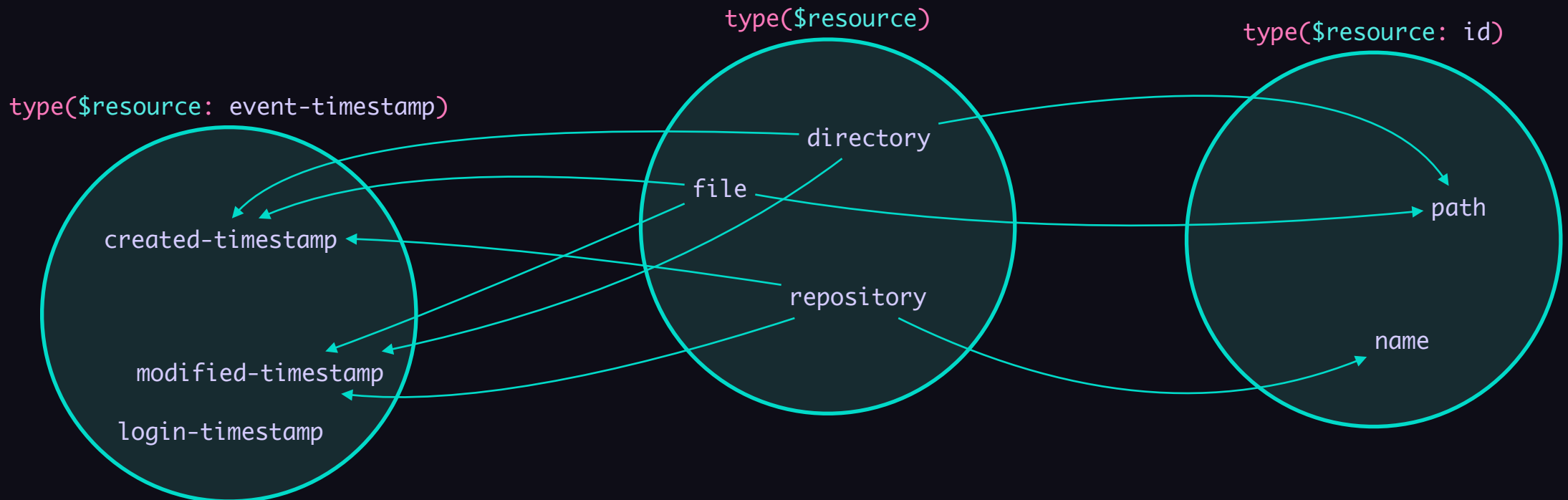


Type inference

"List the ID and event timestamps of every resource."

```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```

```
resource owns modified-timestamp;
file sub resource;
directory sub resource;
repository sub resource;
```

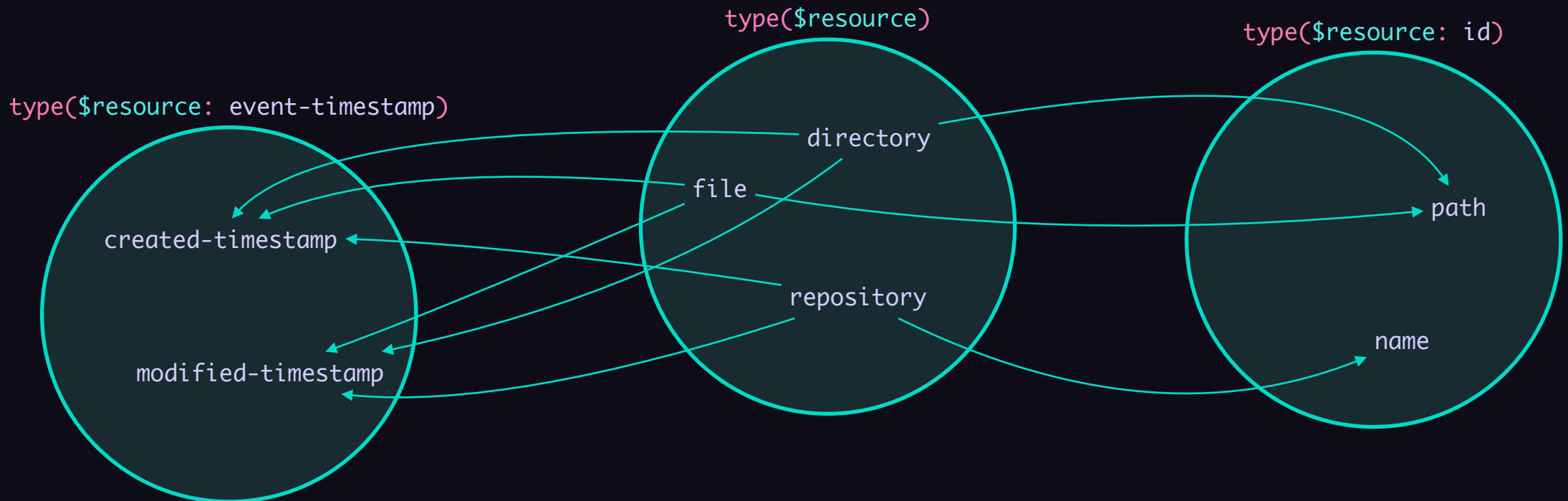


Type inference

"List the ID and event timestamps of every resource."

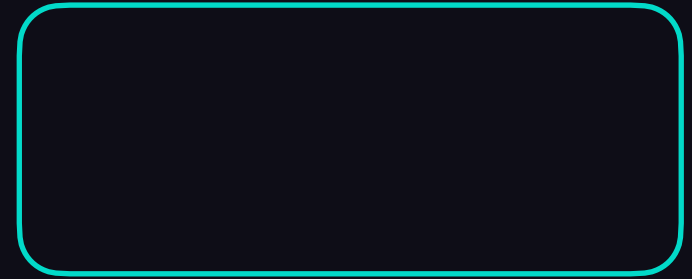


```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```

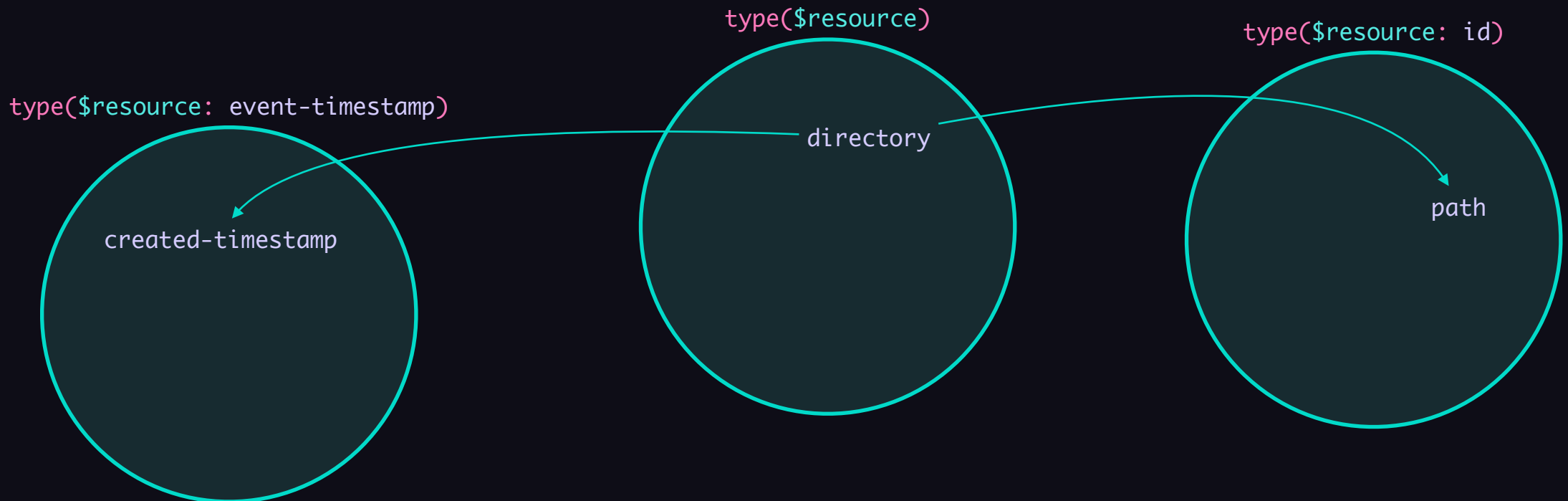


Type inference

"List the ID and event timestamps of every resource."



```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```

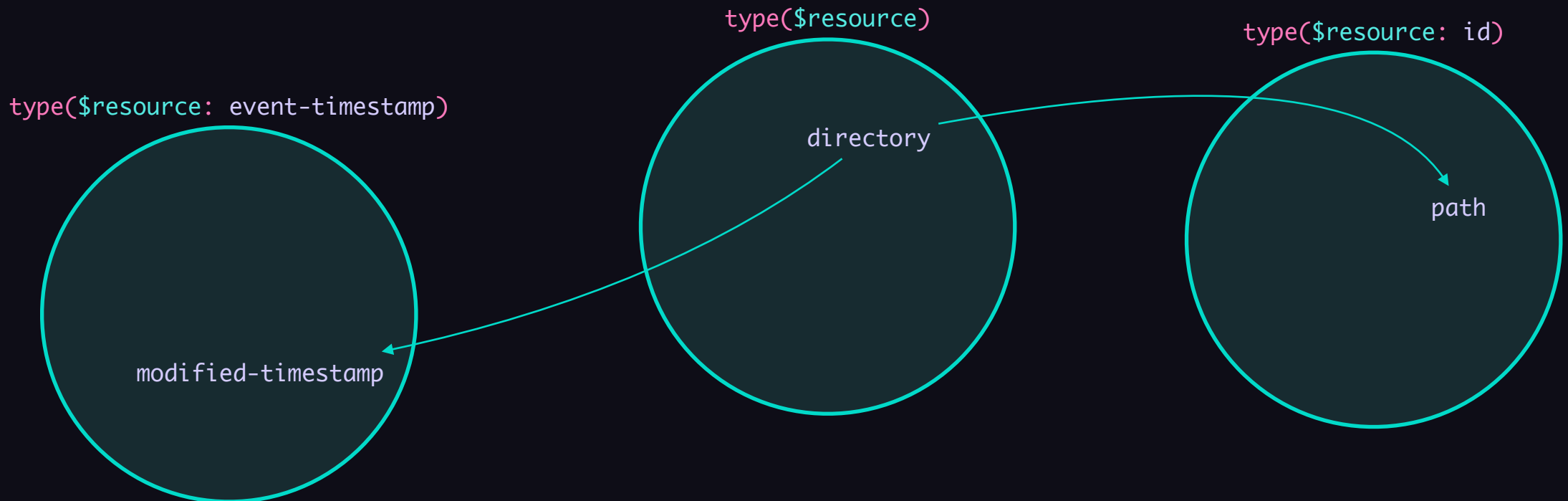


Type inference

"List the ID and event timestamps of every resource."



```
match
  $resource isa resource;
fetch
  $resource: id, event-timestamp;
```

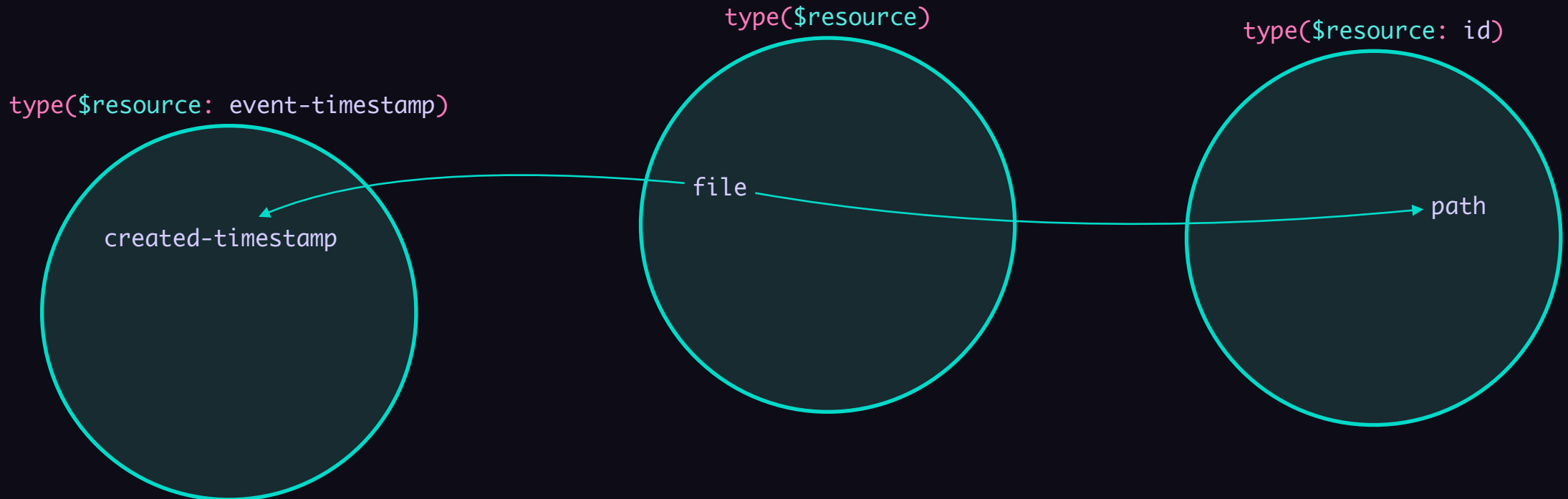


Type inference

"List the ID and event timestamps of every resource."



```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```

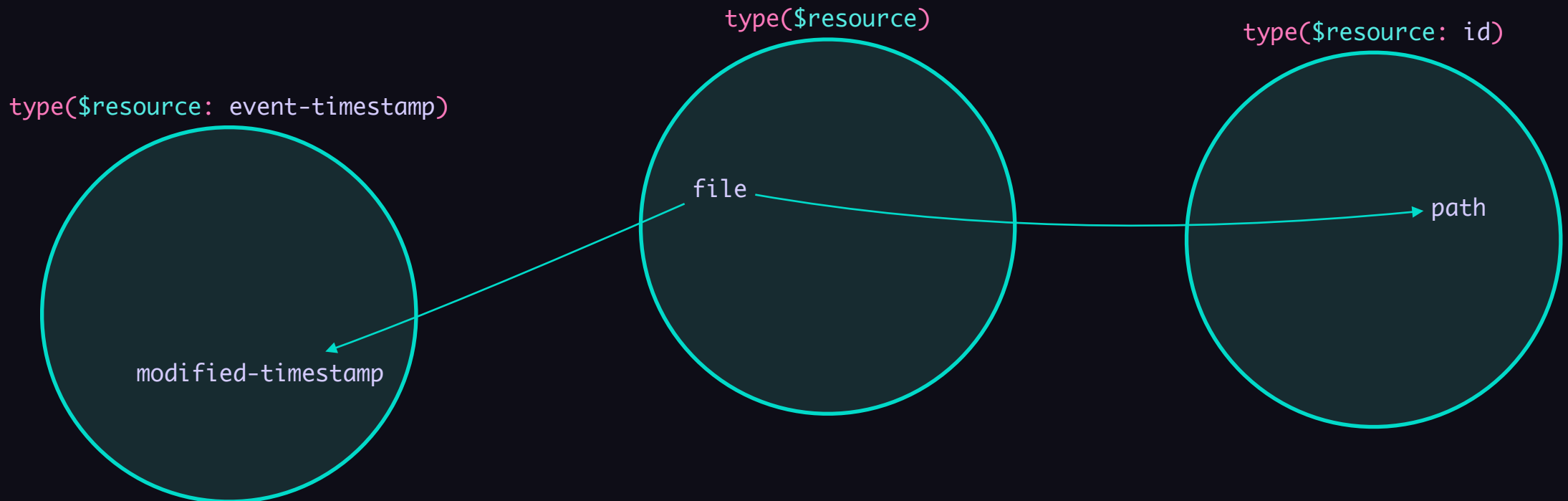


Type inference

"List the ID and event timestamps of every resource."



```
match
  $resource isa resource;
fetch
  $resource: id, event-timestamp;
```

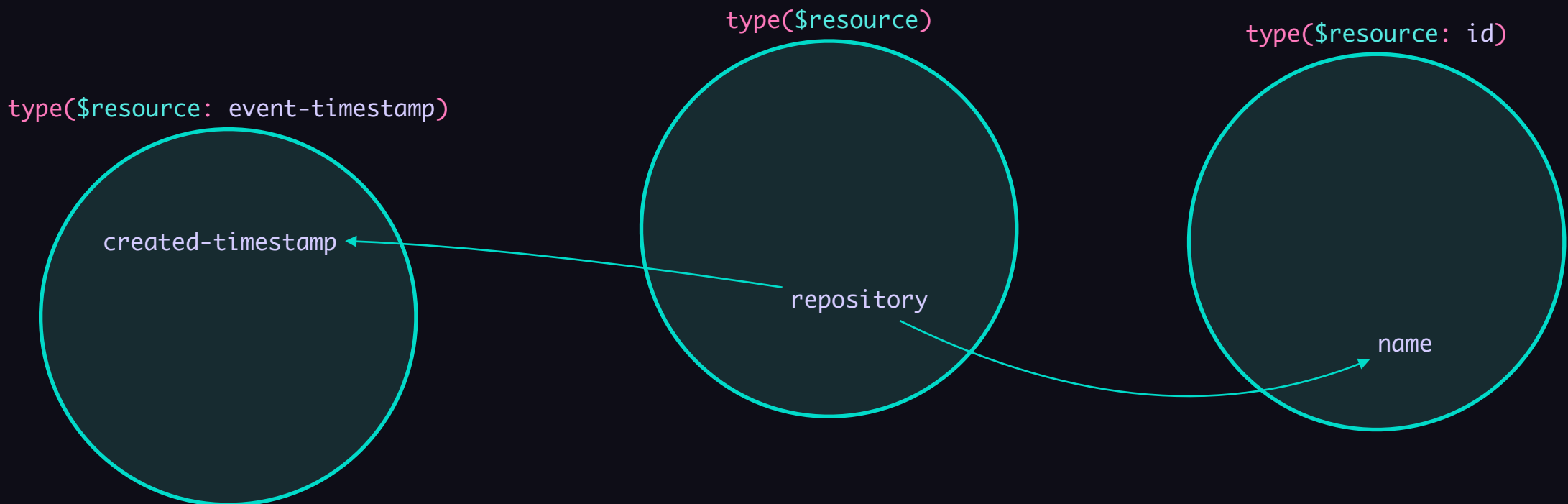


Type inference

"List the ID and event timestamps of every resource."



```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```



Type inference

"List the ID and event timestamps of every resource."



```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```

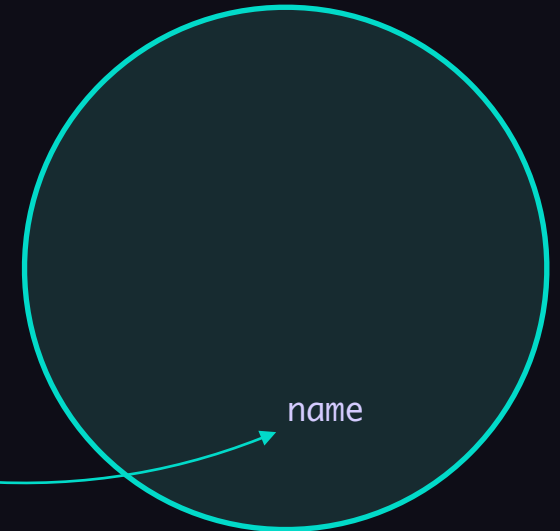
`type($resource: event-timestamp)`



`type($resource)`



`type($resource: id)`



Type inference

"List the ID and event timestamps of every resource."

```
match
$resource isa resource;
fetch
$resource: id, event-timestamp;
```

<code>type(\$resource)</code>	<code>type(\$resource: id)</code>	<code>type(\$resource: event-timestamp)</code>
file	path	created-timestamp
file	path	modified-timestamp
directory	path	created-timestamp
directory	path	modified-timestamp
repository	name	created-timestamp
repository	name	modified-timestamp

Type inference

"List the ID and owner ID of every resource."

```
match
(resource: $resource, resource-owner: $owner) isa resource-ownership;
fetch
$resource: id;
$owner: id;
```

Query returns four variables:

\$resource

\$owner

\$resource: id

\$owner: id

Query has four return types:

type(\$resource)

type(\$owner)

type(\$resource: id)

type(\$owner: id)

Type inference

"List the ID and owner ID of every resource."

```
match  
(resource: $resource, resource-owner: $owner) isa resource-ownership;  
fetch  
$resource: id;  
$owner: id;
```

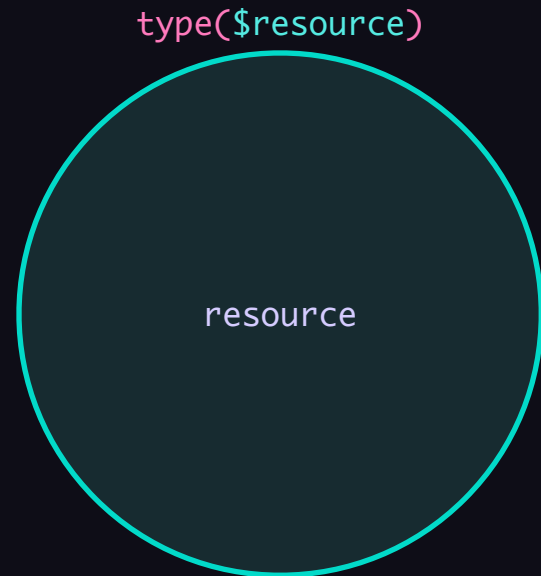
type(\$resource)



Type inference

"List the ID and owner ID of every resource."

```
match  
(resource: $resource, resource-owner: $owner) isa resource-ownership;  
fetch  
$resource: id;  
$owner: id;
```



```
resource plays resource-ownership:resource;
```

Type inference

"List the ID and owner ID of every resource."

```
match
(resource: $resource, resource-owner: $owner) isa resource-ownership;
fetch
$resource: id;
$owner: id;
```



```
file sub resource;  
directory sub resource;  
repository sub resource;
```

Type inference

"List the ID and owner ID of every resource."

```
match
(resource: $resource, resource-owner: $owner) isa resource-ownership;
fetch
$resource: id;
$owner: id;
```



```
resource abstract;
```

Type inference

"List the ID and owner ID of every resource."

```
match  
(resource: $resource, resource-owner: $owner) isa resource-ownership;  
fetch  
$resource: id;  
$owner: id;
```

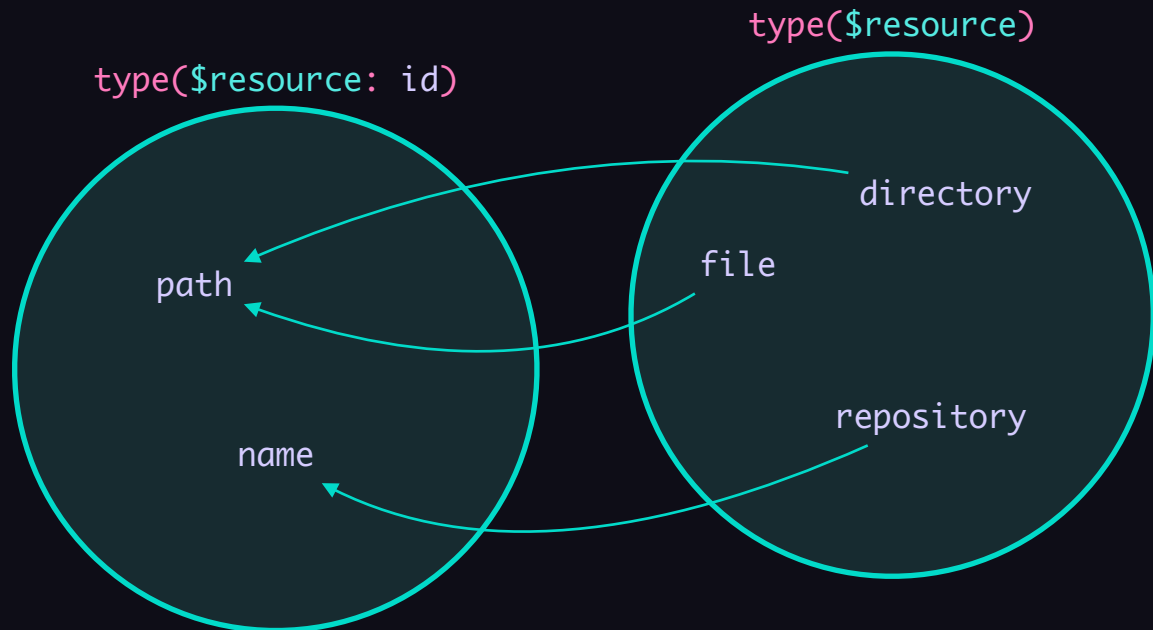


Type inference

"List the ID and owner ID of every resource."

```
match
(resource: $resource, resource-owner: $owner) isa resource-ownership;
fetch
$resource: id;
$owner: id;
```

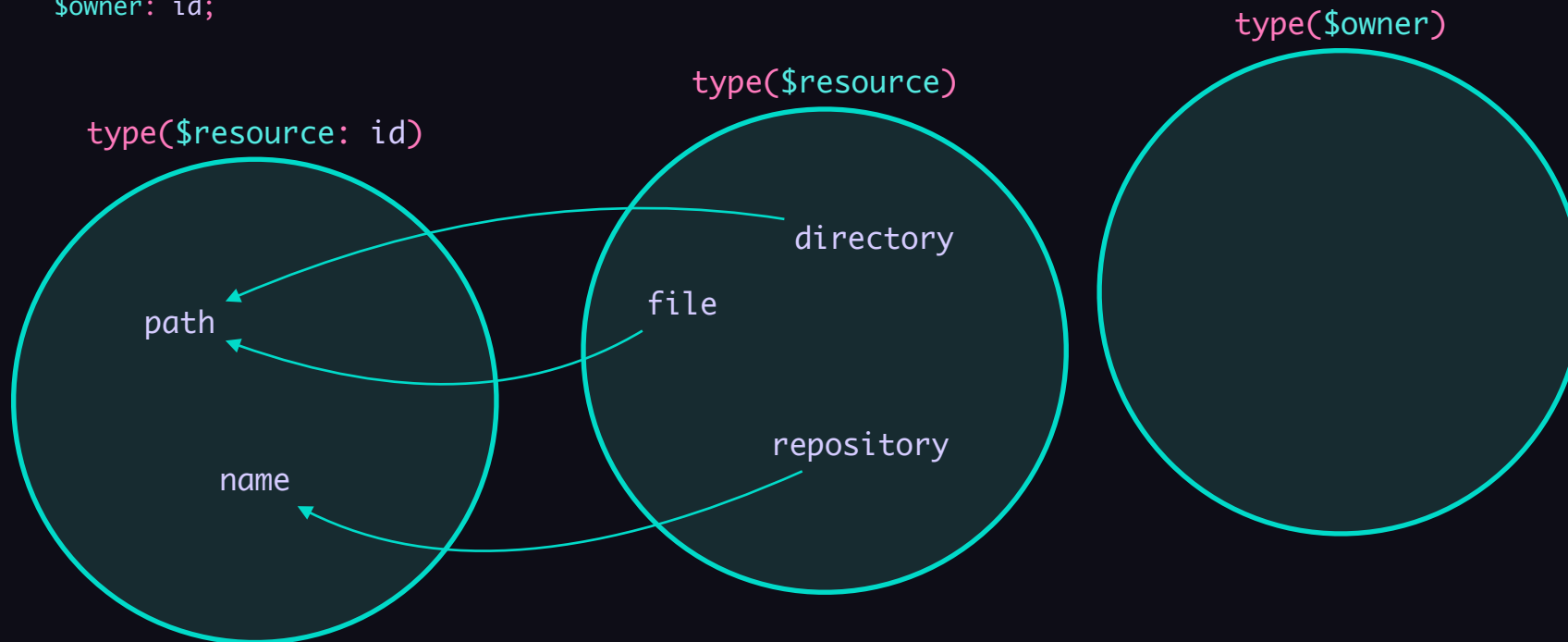
```
email sub id; name sub id;
path sub id; hash sub id;
resource abstract;
file owns path;
directory owns path;
repository owns name;
```



Type inference

"List the ID and owner ID of every resource."

```
match  
(resource: $resource, resource-owner: $owner) isa resource-ownership;  
fetch  
$resource: id;  
$owner: id;
```

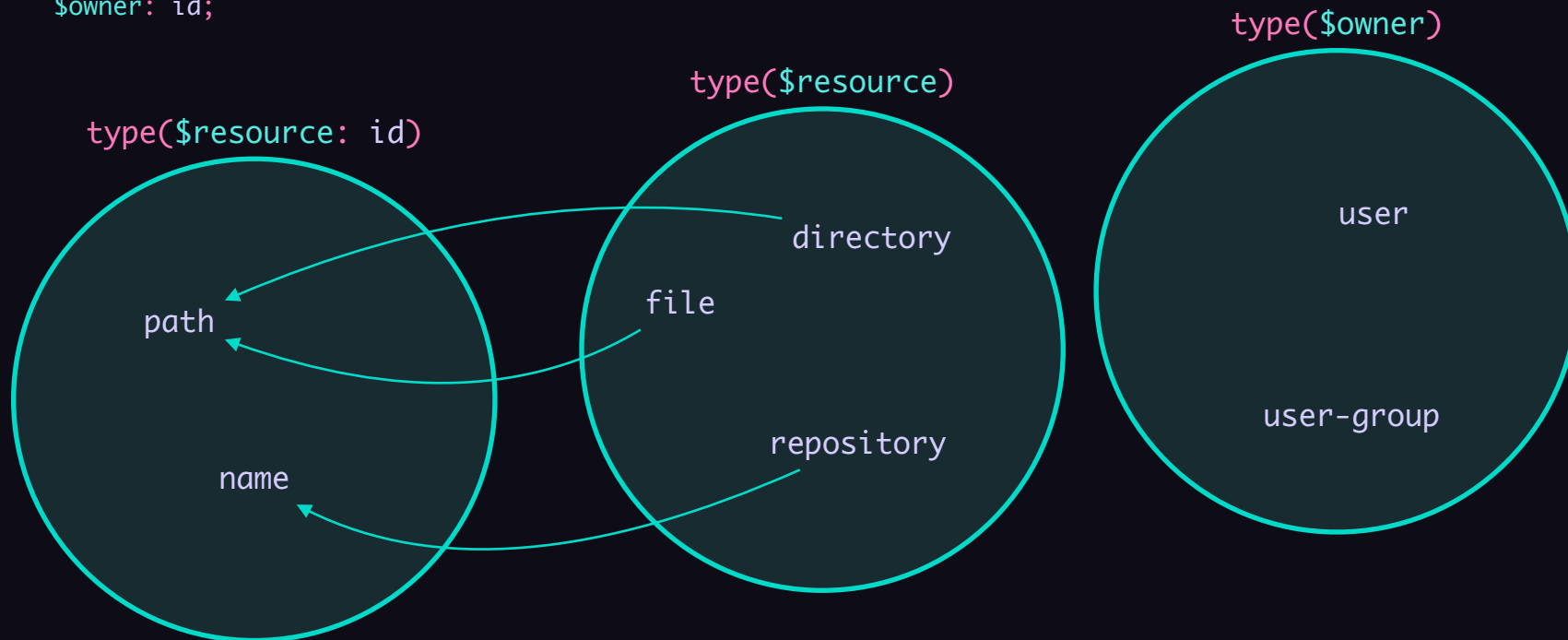


Type inference

"List the ID and owner ID of every resource."

```
user plays resource-ownership:resource-owner;  
user-group plays resource-ownership:resource-owner;
```

```
match  
(resource: $resource, resource-owner: $owner) isa resource-ownership;  
fetch  
$resource: id;  
$owner: id;
```

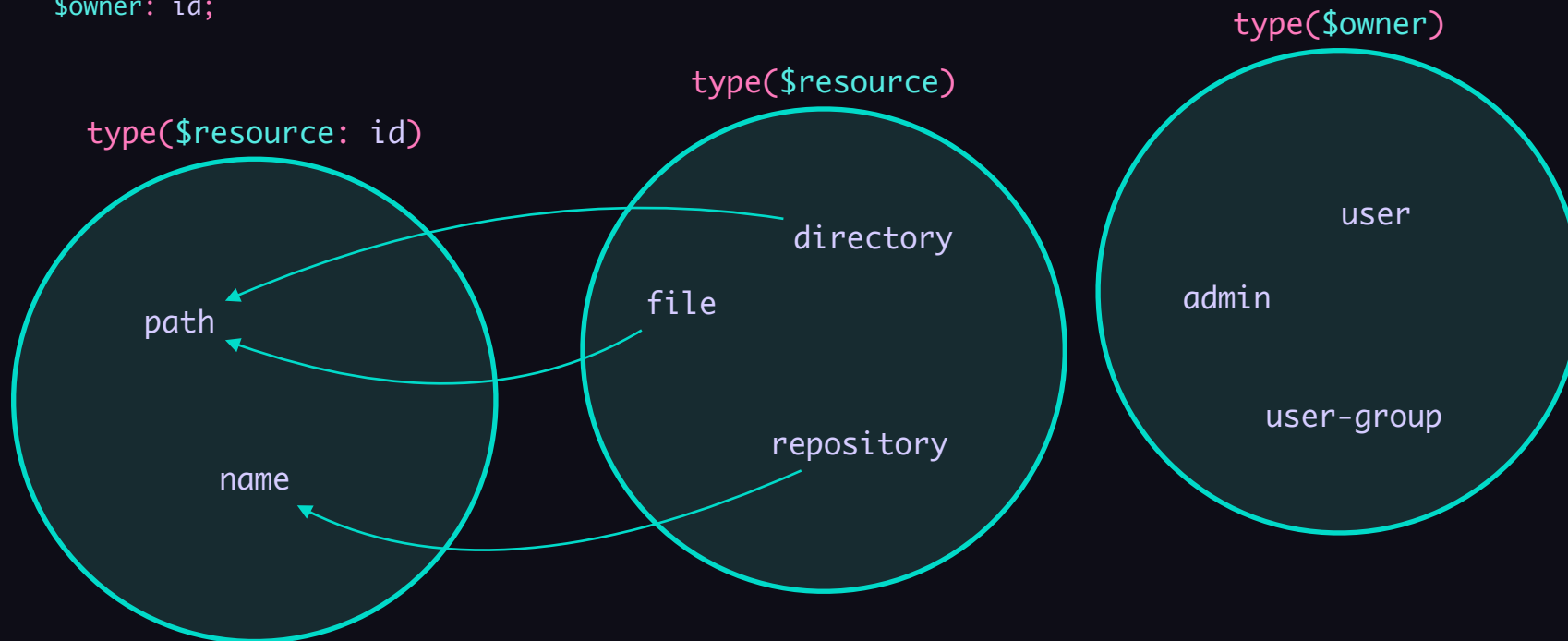


Type inference

"List the ID and owner ID of every resource."

```
match
(resource: $resource, resource-owner: $owner) isa resource-ownership;
fetch
$resource: id;
$owner: id;
```

```
admin sub user;
```

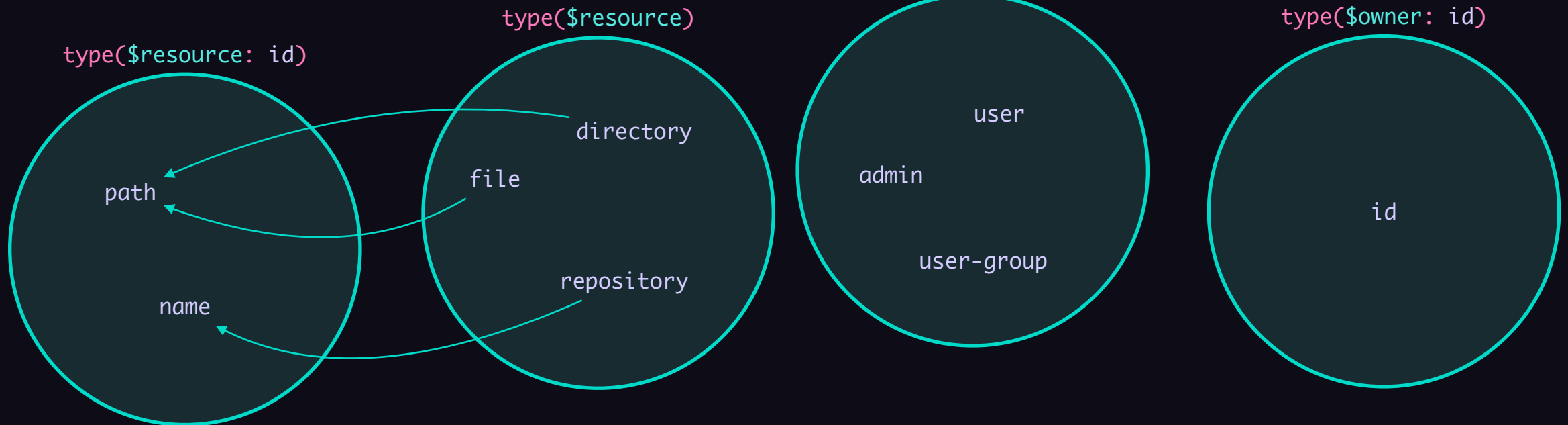


Type inference

"List the ID and owner ID of every resource."



```
match
(resource: $resource, resource-owner: $owner) isa resource-ownership;
fetch
$resource: id;
$owner: id;
```

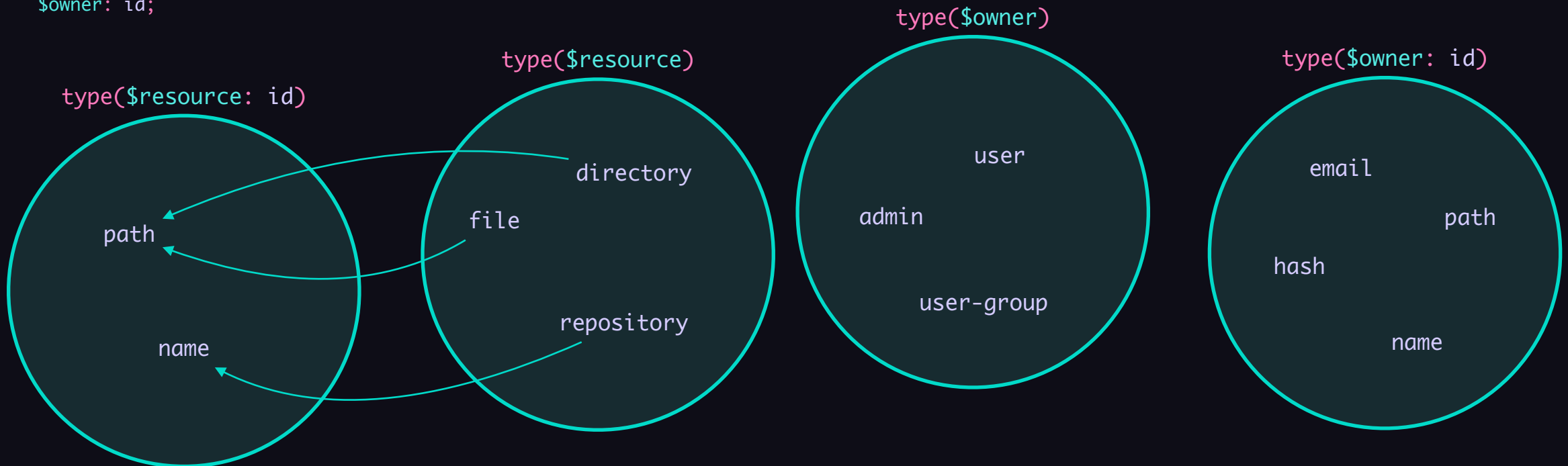


Type inference

"List the ID and owner ID of every resource."

```
email sub id; name sub id;  
path sub id; hash sub id;  
resource abstract;
```

```
match  
(resource: $resource, resource-owner: $owner) isa resource-ownership;  
fetch  
$resource: id;  
$owner: id;
```

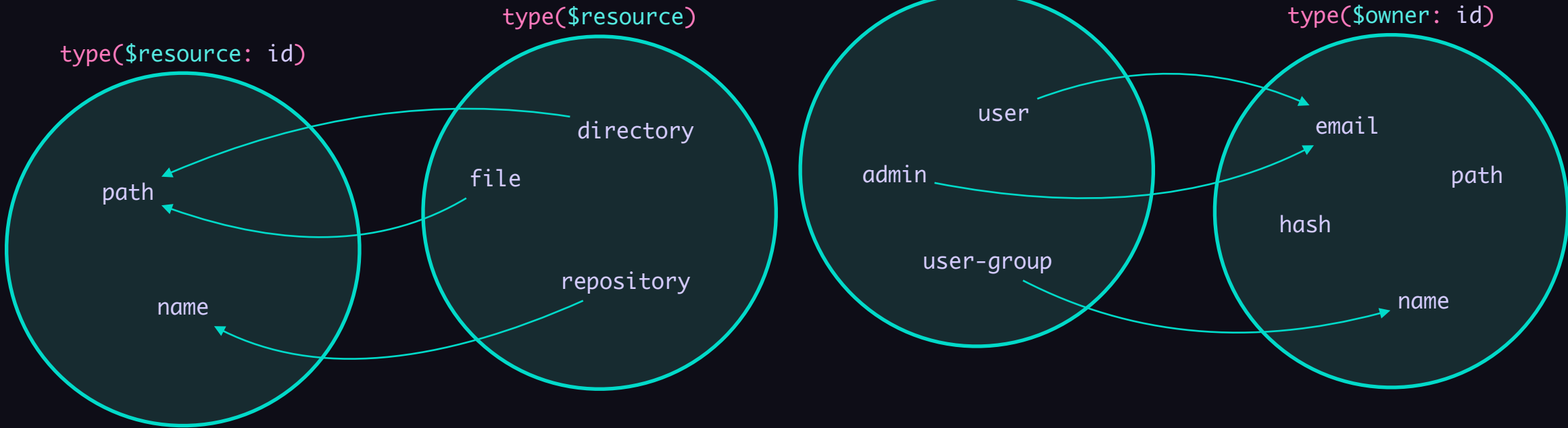


Type inference

"List the ID and owner ID of every resource."

```
user owns email;  
admin sub user;  
user-group owns name;
```

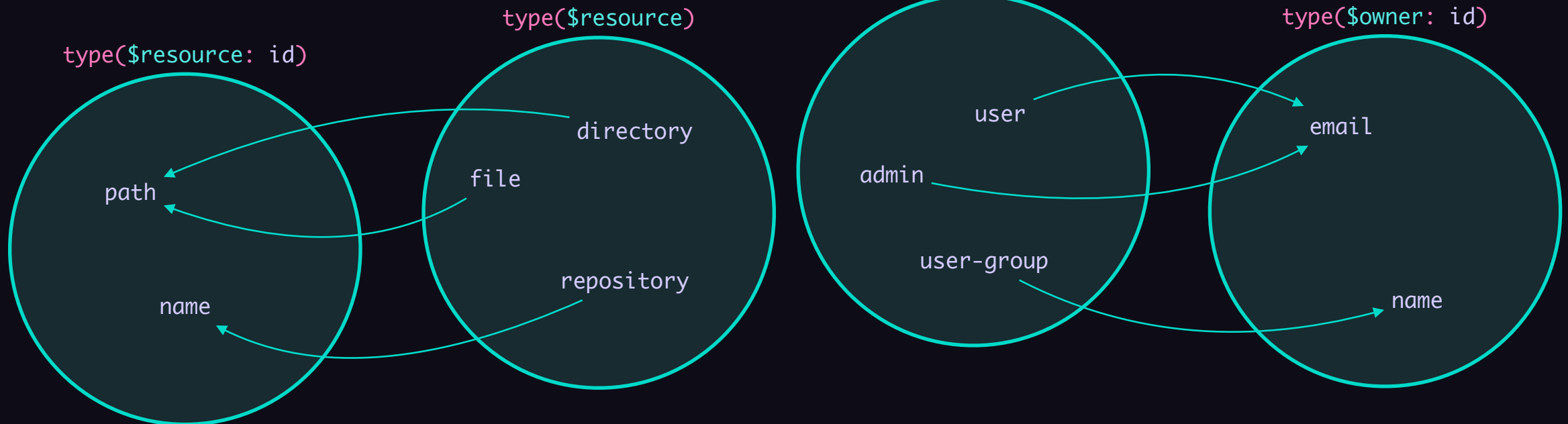
```
match  
(resource: $resource, resource-owner: $owner) isa resource-ownership;  
fetch  
$resource: id;  
$owner: id;
```



Type inference

"List the ID and owner ID of every resource."

```
match
(resource: $resource, resource-owner: $owner) isa resource-ownership;
fetch
$resource: id;
$owner: id;
```

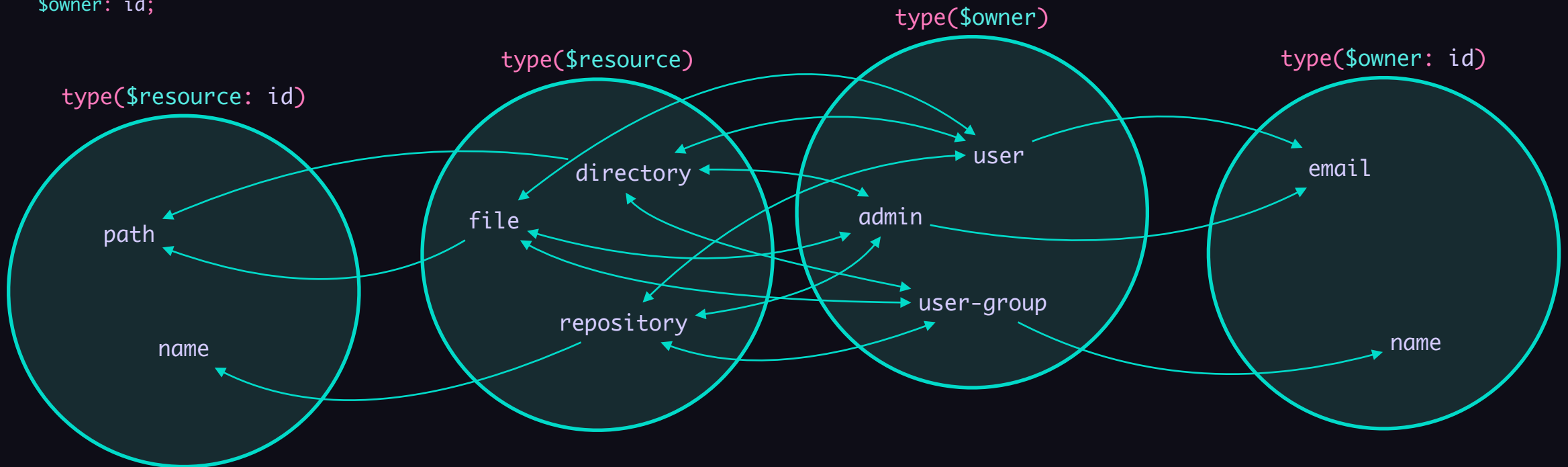


Type inference

"List the ID and owner ID of every resource."



```
match
(resource: $resource, resource-owner: $owner) isa resource-ownership;
fetch
$resource: id;
$owner: id;
```



Type inference

"List the ID and owner ID of every resource."

```
match
(resource: $resource, resource-owner: $owner) isa resource-ownership;
fetch
$resource: id;
$owner: id;
```

type(\$resource)	type(\$resource: id)	type(\$owner)	type(\$owner: id)
file	path	user	email
file	path	admin	email
file	path	user-group	name
directory	path	user	email
directory	path	admin	email
directory	path	user-group	name
repository	name	user	email
repository	name	admin	email
repository	name	user-group	name

Type inference

"List the ID and owner ID of every resource."

```
match
(resource: $resource, resource-owner: $owner) isa resource-ownership;
fetch
$resource: id;
$owner: id;
```

Possible pairs of return types

\$resource	\$owner
file	user
file	admin
file	user-group
directory	user
directory	admin
directory	user-group

type(\$resource)	type(\$resource: id)	type(\$owner)	type(\$owner: id)
file	path	user	email
file	path	admin	email
file	path	user-group	name
directory	path	user	email
directory	path	admin	email
directory	path	user-group	name
repository	name	user	email
repository	name	admin	email
repository	name	user-group	name

Semantic validation

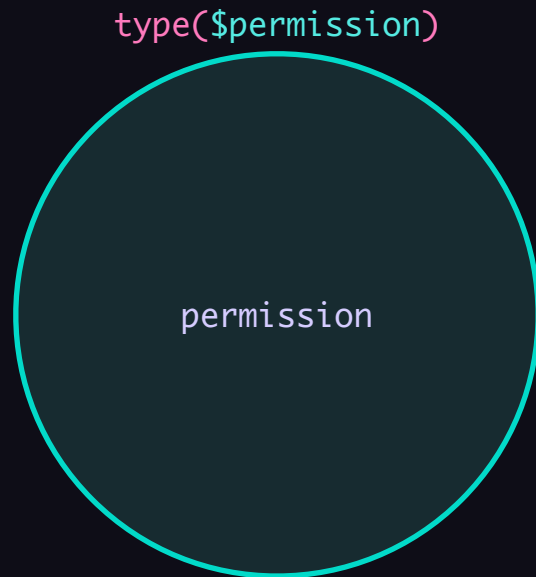
"List the ID of every permission and its roleplayers."

```
match
$permission (subject: $subject, object: $object, access: $access) isa permission;
fetch
$permission: id;
$subject: id;
$object: id;
$access: id;
```

Semantic validation

"List the ID of every permission and its roleplayers."

```
match
$permission (subject: $subject, object: $object, access: $access) isa permission;
fetch
$permission: id;
$subject: id;
$object: id;
$access: id;
```



Semantic validation

"List the ID of every permission and its roleplayers."

```
match
$permission (subject: $subject, object: $object, access: $access) isa permission;
fetch
$permission: id;
$subject: id;
$object: id;
$access: id;
```

type(\$permission)



type(\$permission: id)



Semantic validation

"List the ID of every permission and its roleplayers."

```
email sub id; name sub id;  
path sub id; hash sub id;  
resource abstract;
```

```
match  
$permission (subject: $subject, object: $object, access: $access) isa permission;  
fetch  
$permission: id;  
$subject: id;  
$object: id;  
$access: id;
```

type(\$permission)



type(\$permission: id)



Semantic validation

"List the ID of every permission and its roleplayers."

```
match
$permission (subject: $subject, object: $object, access: $access) isa permission;
fetch
$permission: id;
$subject: id;
$object: id;
$access: id;
```

permission does not own
email, name, path, or hash

type(\$permission)



type(\$permission: id)



Semantic validation

"List the ID of every permission and its roleplayers."

```
match
$permission (subject: $subject, object: $object, access: $access) isa permission;
fetch
$permission: id;
$subject: id;
$object: id;
$access: id;
```

permission does not own
email, name, path, or hash



Semantic validation

"List the ID of every permission and its roleplayers."

```
match
$permission (subject: $subject, object: $object, access: $access) isa permission;
fetch
$permission: id;
$subject: id;
$object: id;
$access: id;
```


```
## Error> [CXN05] The transaction is closed because of the error(s):
[PRO06] Invalid projection operation: Projection from '$permission' to attribute type 'id' is illegal, since '$permission' could be of
type 'permission' which does not own the attribute type or any of its subtypes.
## Terminated
```

Semantic validation

"List the login timestamps of every group owner."

```
match
(group: $group, group-owner: $owner) isa group-ownership;
$login (subject: $group) isa login-event;
fetch
$group: name;
$login: login-timestamp, success;
```

\$group here should be \$owner



Semantic validation

"List the login timestamps of every group owner."

```
match
(group: $group, group-owner: $owner) isa group-ownership;
$login (subject: $group) isa login-event;
fetch
$group: name;
$login: login-timestamp, success;
```

```
user-group plays group-ownership:group;
```

type(\$group)

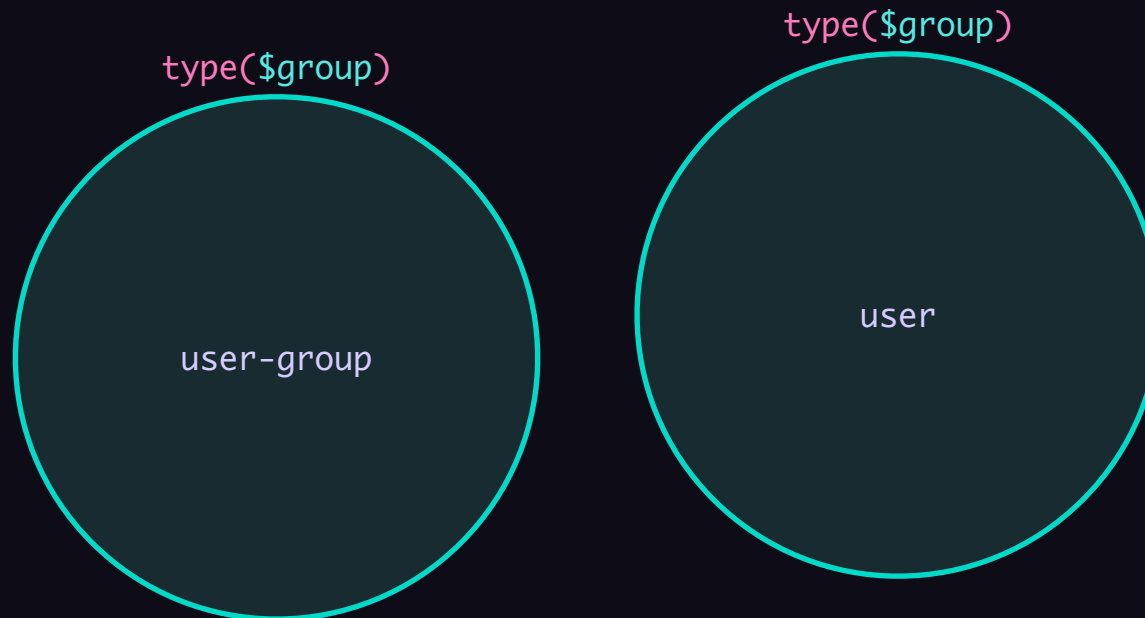
user-group

Semantic validation

"List the login timestamps of every group owner."

```
user plays login-event:subject;
```

```
match
(group: $group, group-owner: $owner) isa group-ownership;
$login (subject: $group) isa login-event;
fetch
$group: name;
$login: login-timestamp, success;
```

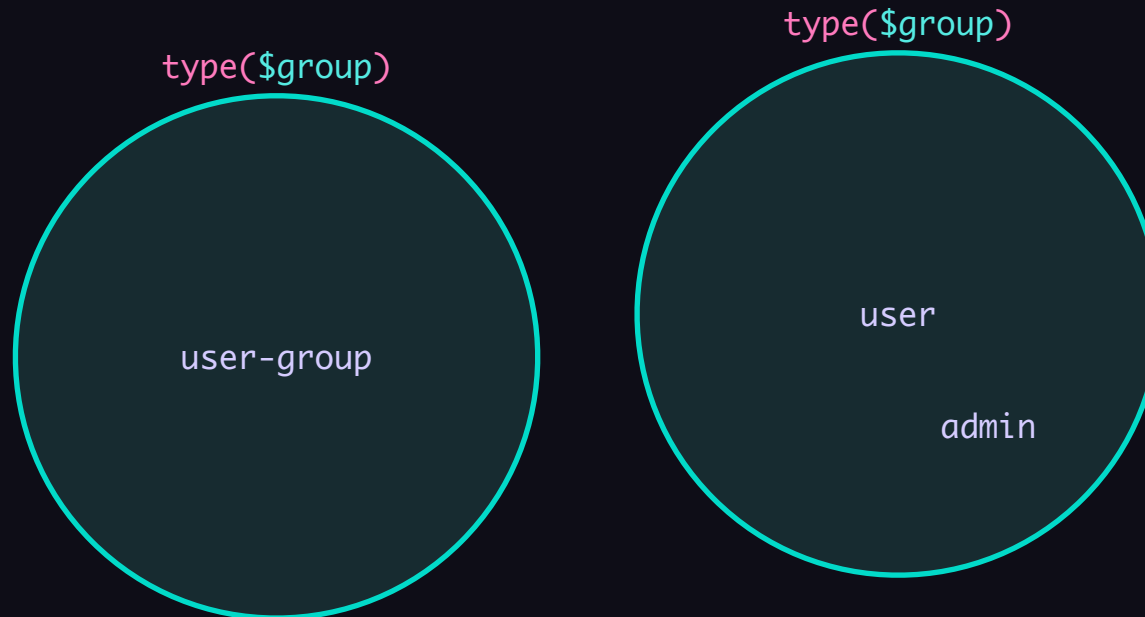


Semantic validation

"List the login timestamps of every group owner."

```
match
(group: $group, group-owner: $owner) isa group-ownership;
$login (subject: $group) isa login-event;
fetch
$group: name;
$login: login-timestamp, success;
```

```
admin sub user;
```

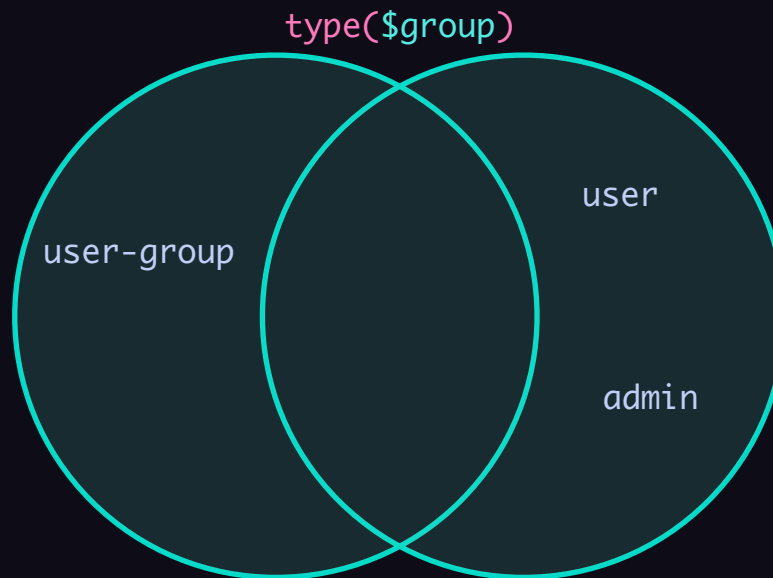


Semantic validation



"List the login timestamps of every group owner."

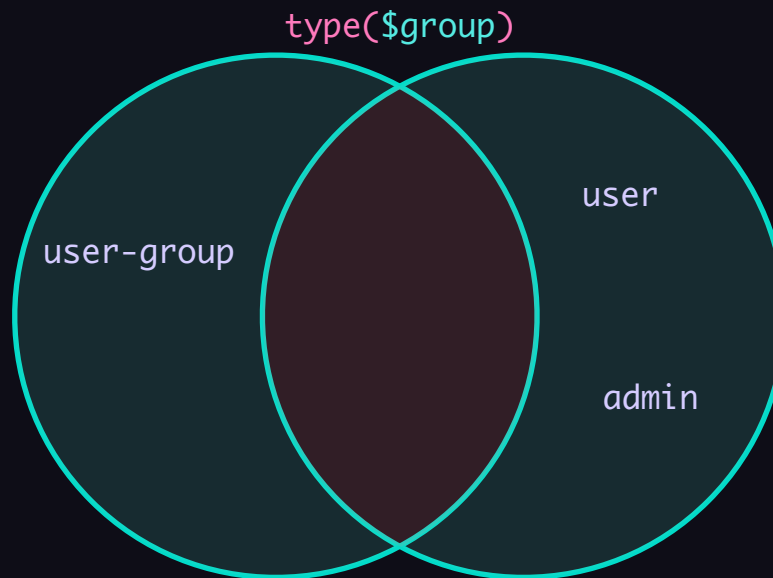
```
match
(group: $group, group-owner: $owner) isa group-ownership;
$login (subject: $group) isa login-event;
fetch
$group: name;
$login: login-timestamp, success;
```



Semantic validation

"List the login timestamps of every group owner."

```
match
(group: $group, group-owner: $owner) isa group-ownership;
$login (subject: $group) isa login-event;
fetch
$group: name;
$login: login-timestamp, success;
```



Semantic validation

"List the login timestamps of every group owner."

```
match
(group: $group, group-owner: $owner) isa group-ownership;
$login (subject: $group) isa login-event;
fetch
$group: name;
$login: login-timestamp, success;

## Error> [CXN05] The transaction is closed because of the error(s):
[QRY16] Invalid Query Pattern: Could not infer compatible types for the match pattern.
## Terminated
```

Strong type system

- Type inference resolves queries against the schema.
- Semantically invalid queries throw errors.

Symbolic reasoning

Integrated application logic

Defining rules

- Simple rules.
- Rule branching.
- Rule chaining.
- Rule recursion.

Simple rules

"List the most recent modified timestamp for each resource."

```
match
$resource isa resource, has modified-timestamp $t;
not {
  $resource has modified-timestamp $t-2;
  $t-2 > $t;
};
fetch
$resource: id;
$t;
```

```
{
  "resource": {
    "id": [ { "value": "/vaticle/engineering/tools", "value_type": "string", "type": { "label": "path", "root": "attribute" } } ],
    "type": { "label": "directory", "root": "entity" }
  },
  "t": { "value": "2023-11-19T18:32:25.000", "value_type": "datetime", "type": { "label": "modified-timestamp", "root": "attribute" } }
}
```

Simple rules

"List the most recent modified timestamp for each resource."

```
match
  $resource isa resource, has modified-timestamp $t;
not {
  $resource has modified-timestamp $t-2;
  $t-2 > $t;
};
fetch
  $resource: id;
  $t;
```

```
define
  last-modified sub attribute, value datetime;
  resource owns last-modified;

rule resource-last-modified:
  when {
    ...
  } then {
    ...
  };
```

Simple rules

"List the most recent modified timestamp for each resource."

```
match
{
  $resource isa resource, has modified-timestamp $t;
  not {
    $resource has modified-timestamp $t-2;
    $t-2 > $t;
  };
  fetch
  $resource: id;
  $t;
}
```

```
define
last-modified sub attribute, value datetime;
resource owns last-modified;

rule resource-last-modified:
  when {
    # The original query pattern
    $resource isa resource, has modified-timestamp $t;
    not {
      $resource has modified-timestamp $t-2;
      $t-2 > $t;
    };
    ...
  } then {
    ...
  };
```

Simple rules

"List the most recent modified timestamp for each resource."

```
match
{
  $resource isa resource, has modified-timestamp $t;
  not {
    $resource has modified-timestamp $t-2;
    $t-2 > $t;
  };
  fetch
  $resource: id;
  $t;
}
```

```
define
last-modified sub attribute, value datetime;
resource owns last-modified;

rule resource-last-modified:
  when {
    # The original query pattern
    $resource isa resource, has modified-timestamp $t;
    not {
      $resource has modified-timestamp $t-2;
      $t-2 > $t;
    };
    # Copy the value of $t
    ?last-t = $t;
  } then {
    ...
  };
```


Simple rules

"List the most recent modified timestamp for each resource."

```
match
{
  $resource isa resource, has modified-timestamp $t;
  not {
    $resource has modified-timestamp $t-2;
    $t-2 > $t;
  };
  fetch
  $resource: id;
  $t;
```

```
define
  last-modified sub attribute, value datetime;
  resource owns last-modified;

rule resource-last-modified:
  when {
    # The original query pattern
    $resource isa resource, has modified-timestamp $t;
    not {
      $resource has modified-timestamp $t-2;
      $t-2 > $t;
    };
    # Copy the value of $t
    ?last-t = $t;
  } then {
    # Generate a new attribute with the same value
    $resource has last-modified ?last-t;
  };
```

Simple rules

"List the most recent modified timestamp for each resource."

```
match
{
  $resource isa resource, has modified-timestamp $t;
  not {
    $resource has modified-timestamp $t-2;
    $t-2 > $t;
  };
  fetch
  $resource: id;
  $t;
}
```

define

```
last-modified sub attribute, value datetime;
resource owns last-modified;
```

rule resource-last-modified:

when {

The original query pattern

```
$resource isa resource, has modified-timestamp $t;
not {
  $resource has modified-timestamp $t-2;
  $t-2 > $t;
};
```

Copy the value of \$t

```
?last-t = $t;
```

} then {

Generate a new attribute with the same value

```
$resource has last-modified ?last-t;
```

```
};
```

```
match
  $resource isa resource, has last-modified $t;
  fetch
  $resource: id;
  $t;
```

Simple rules

"List the most recent modified timestamp for each resource."

```
match
$resource isa resource, has modified-timestamp $t;
not {
  $resource has modified-timestamp $t-2;
  $t-2 > $t;
};
fetch
$resource: id;
$t;
```

```
match
$resource isa resource, has last-modified $t;
fetch
$resource: id;
$t;
```

Simple rules

"List the most recent modified timestamp for each resource."

```
match
$resource isa resource, has modified-timestamp $t;
not {
  $resource has modified-timestamp $t-2;
  $t-2 > $t;
};
fetch
$resource: id;
$t;
```

```
match
$resource isa resource, has last-modified $t;
fetch
$resource: id;
$t;
```

```
{
  "resource": {
    "id": [ { "value": "/vaticle/engineering/tools", "value_type": "string", "type": { "label": "path", "root": "attribute" } } ],
    "type": { "label": "directory", "root": "entity" }
  },
  "t": { "value": "2023-11-19T18:32:25.000", "value_type": "datetime", "type": { "label": "modified-timestamp", "root": "attribute" } }
}
```

Simple rules

"List the most recent modified timestamp for each resource."

```
match
  $resource isa resource, has modified-timestamp $t;
not {
  $resource has modified-timestamp $t-2;
  $t-2 > $t;
};
fetch
  $resource: id;
  $t;
```

```
match
  $resource isa resource, has last-modified $t;
fetch
  $resource: id;
  $t;
```

```
{
  "resource": {
    "id": [ { "value": "/vaticle/engineering/tools", "value_type": "string", "type": { "label": "path", "root": "attribute" } } ],
    "type": { "label": "directory", "root": "entity" }
  },
  "t": { "value": "2023-11-19T18:32:25.000", "value_type": "datetime", "type": { "label": "last-modified", "root": "attribute" } }
}
```

An instance of last-modified is returned instead of modified-timestamp, with the same value

Rule branching

"List the most recent modified timestamp for each resource."

```
match
$resource isa resource, has last-modified $t;
fetch
$resource: id;
$t;
```

```
rule resource-last-modified:
  when {
    $resource isa resource, has modified-timestamp $t;
    not {
      $resource has modified-timestamp $t-2;
      $t-2 > $t;
    };
    ?last-t = $t;
  } then {
    $resource has last-modified ?last-t;
  };
```

Rule branching

"List the most recent modified timestamp for each resource."

```
match
$resource isa resource, has last-modified $t;
fetch
$resource: id;
$t;
```

```
rule resource-last-modified:
  when {
    $resource isa resource, has modified-timestamp $t;
    not {
      $resource has modified-timestamp $t-2;
      $t-2 > $t;
    };
    ?last-t = $t;
  } then {
    $resource has last-modified ?last-t;
  };
```

If there are no instances of modified-timestamp,
then no last-modified instance will be generated

Rule branching

"List the most recent modified timestamp for each resource."

```
match
$resource isa resource, has last-modified $t;
fetch
$resource: id;
$t;

define

rule implicit-resource-last-modified:
  when {
    $resource isa resource, has created-timestamp $t;
    not { $resource has modified-timestamp $t-2; };
    ?last-t = $t;
  } then {
    $resource has last-modified ?last-t;
  };
```

```
rule resource-last-modified:
  when {
    $resource isa resource, has modified-timestamp $t;
    not {
      $resource has modified-timestamp $t-2;
      $t-2 > $t;
    };
    ?last-t = $t;
  } then {
    $resource has last-modified ?last-t;
  };
```


Rule branching

"List the most recent modified timestamp for each resource."

```
match
$resource isa resource, has last-modified $t;
fetch
$resource: id;
$t;
```

define

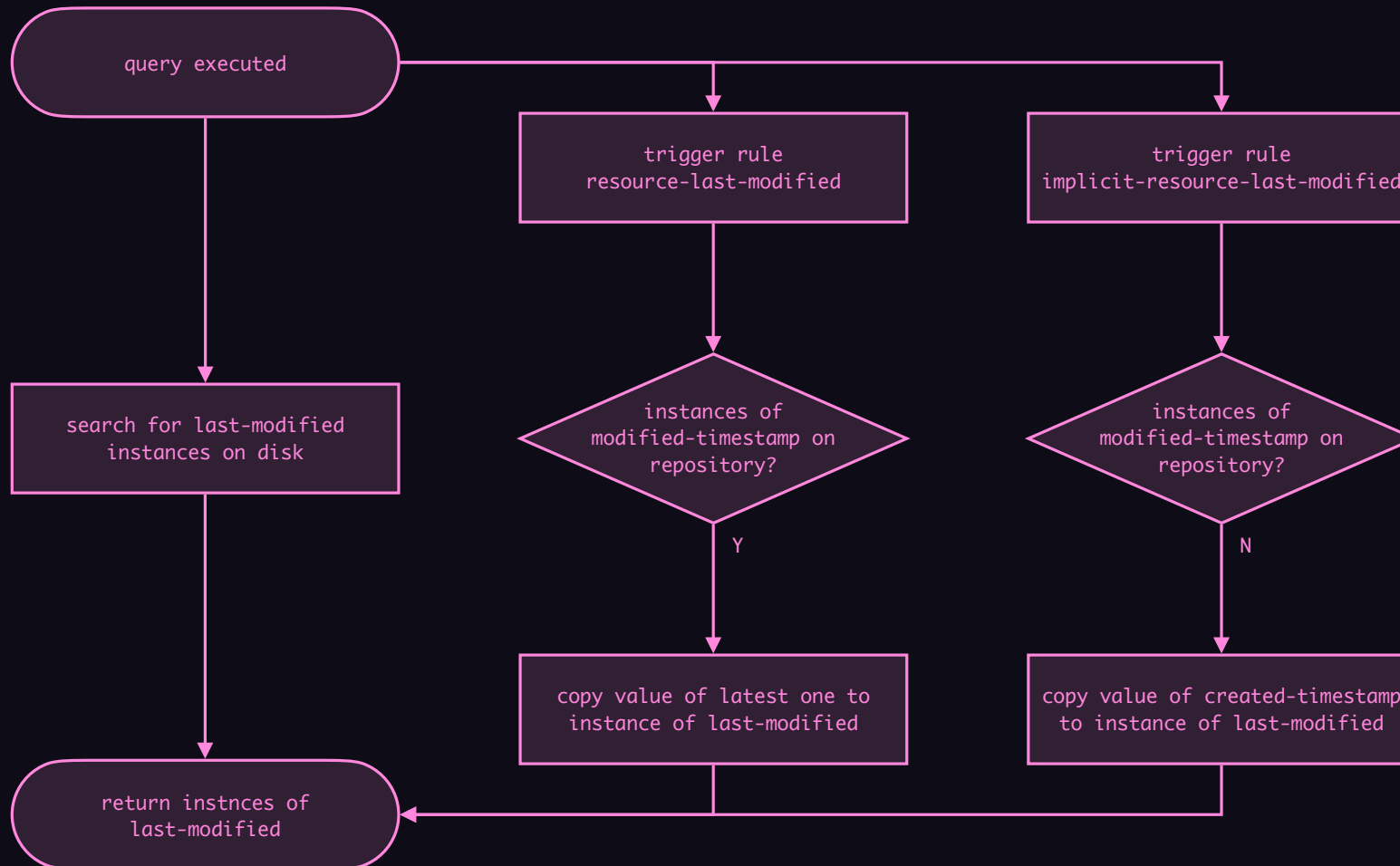
```
rule implicit-resource-last-modified:
  when {
    $resource isa resource, has created-timestamp $t;
    not { $resource has modified-timestamp $t-2; };
    ?last-t = $t;
  } then {
    $resource has last-modified ?last-t;
  };
```

```
rule resource-last-modified:
  when {
    $resource isa resource, has modified-timestamp $t;
    not {
      $resource has modified-timestamp $t-2;
      $t-2 > $t;
    };
    ?last-t = $t;
  } then {
    $resource has last-modified ?last-t;
  };
```

Rule conditions are mutually exclusive

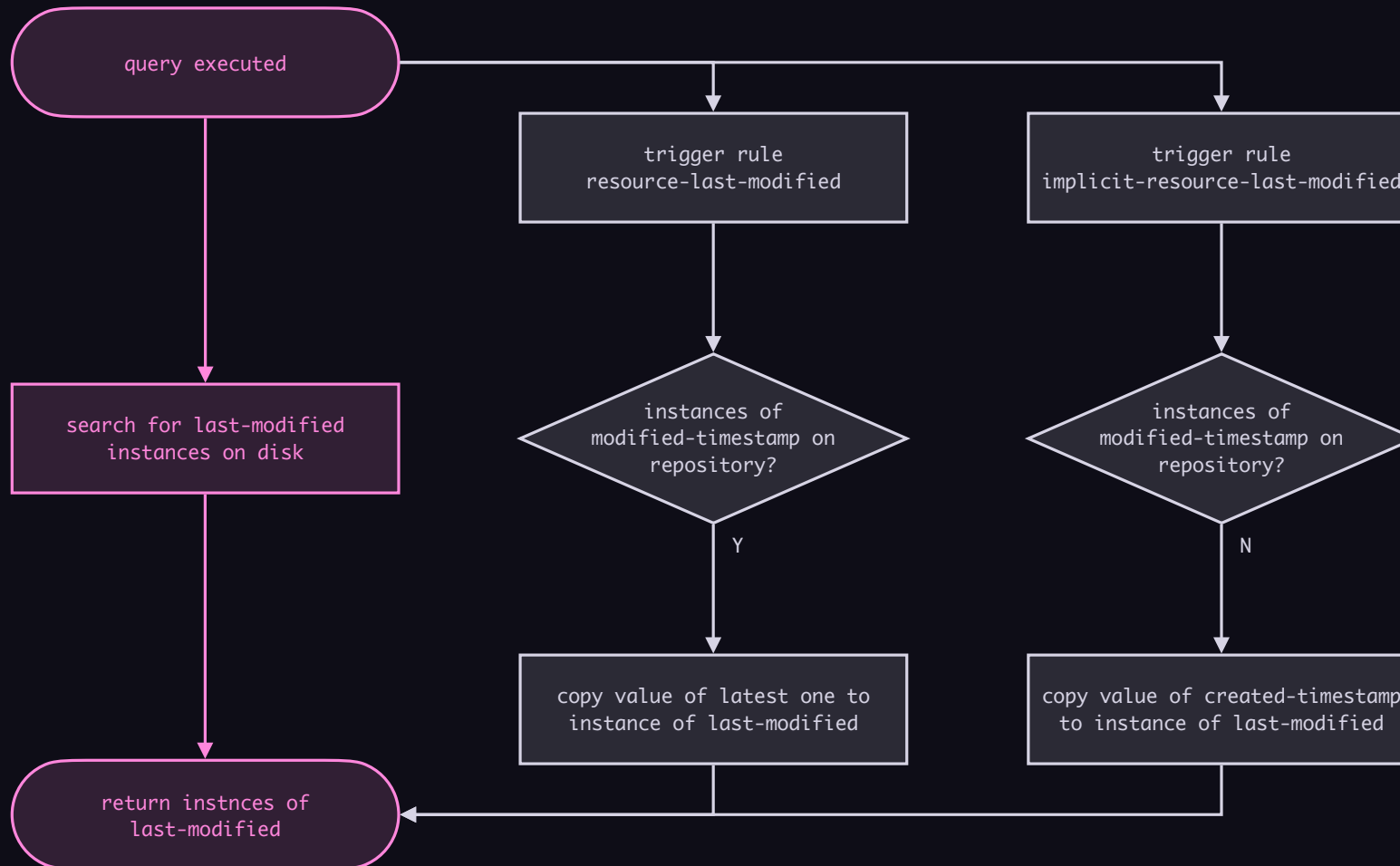
Rule branching

"List the most recent modified timestamp for each resource."



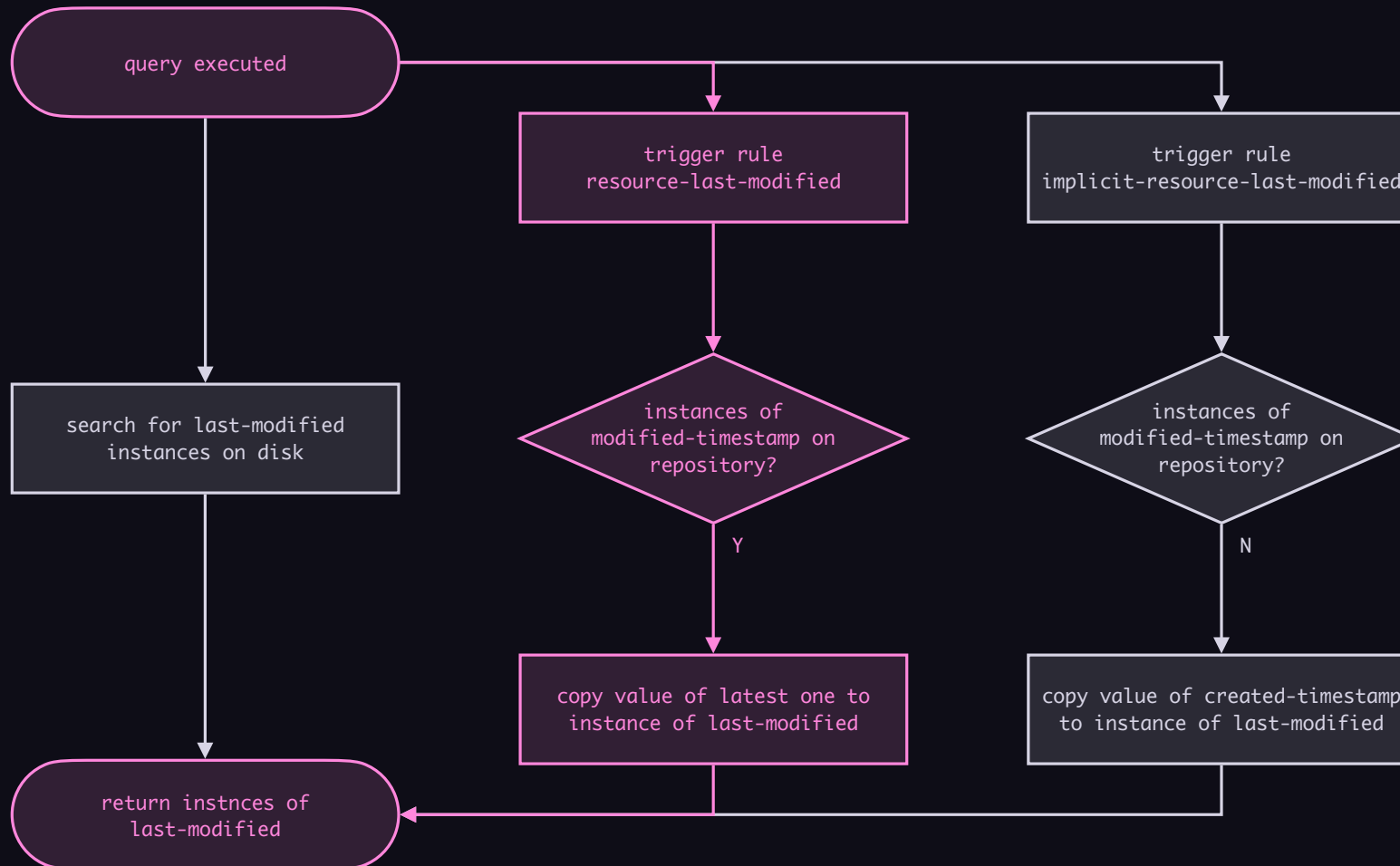
Rule branching

"List the most recent modified timestamp for each resource."



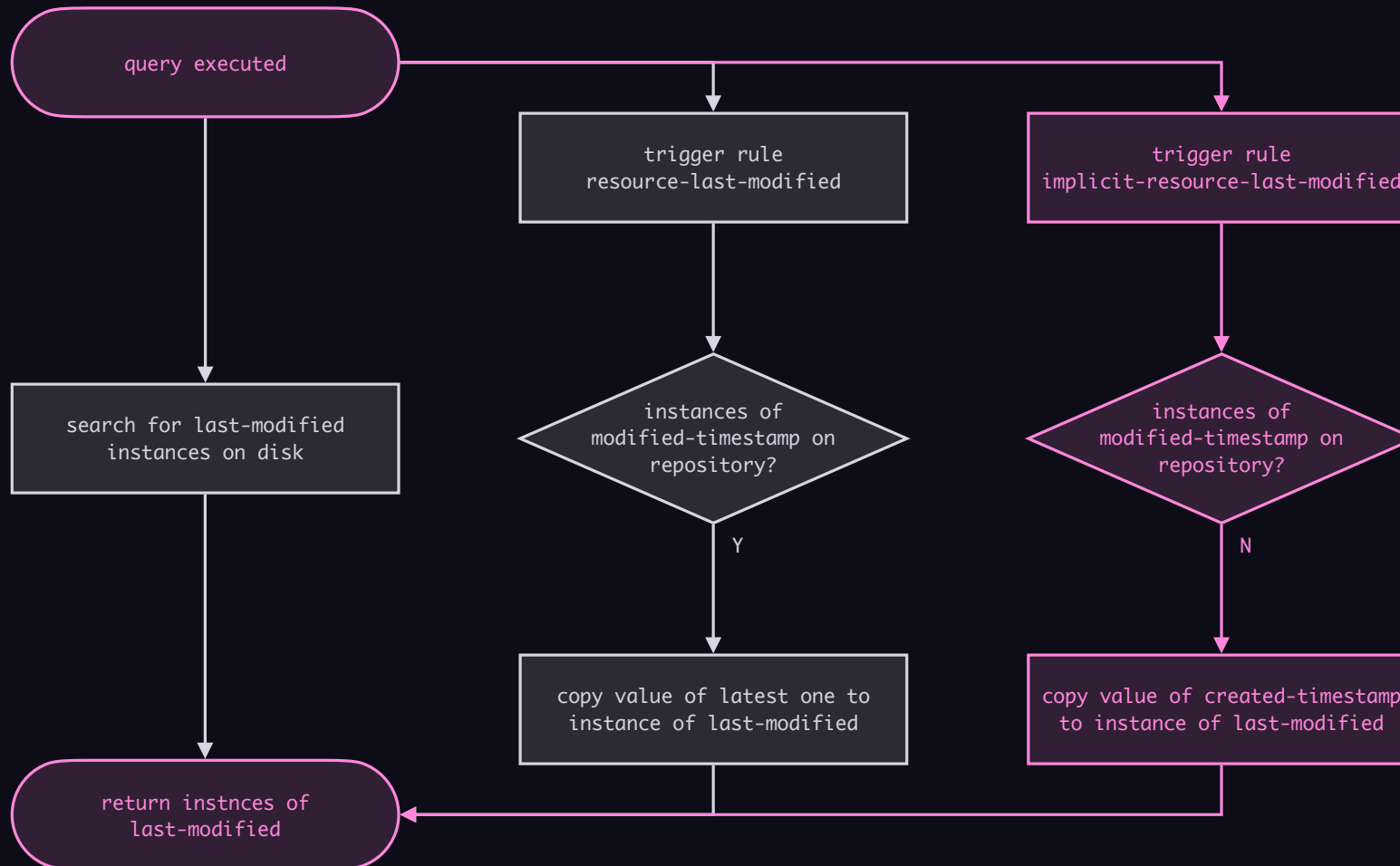
Rule branching

"List the most recent modified timestamp for each resource."



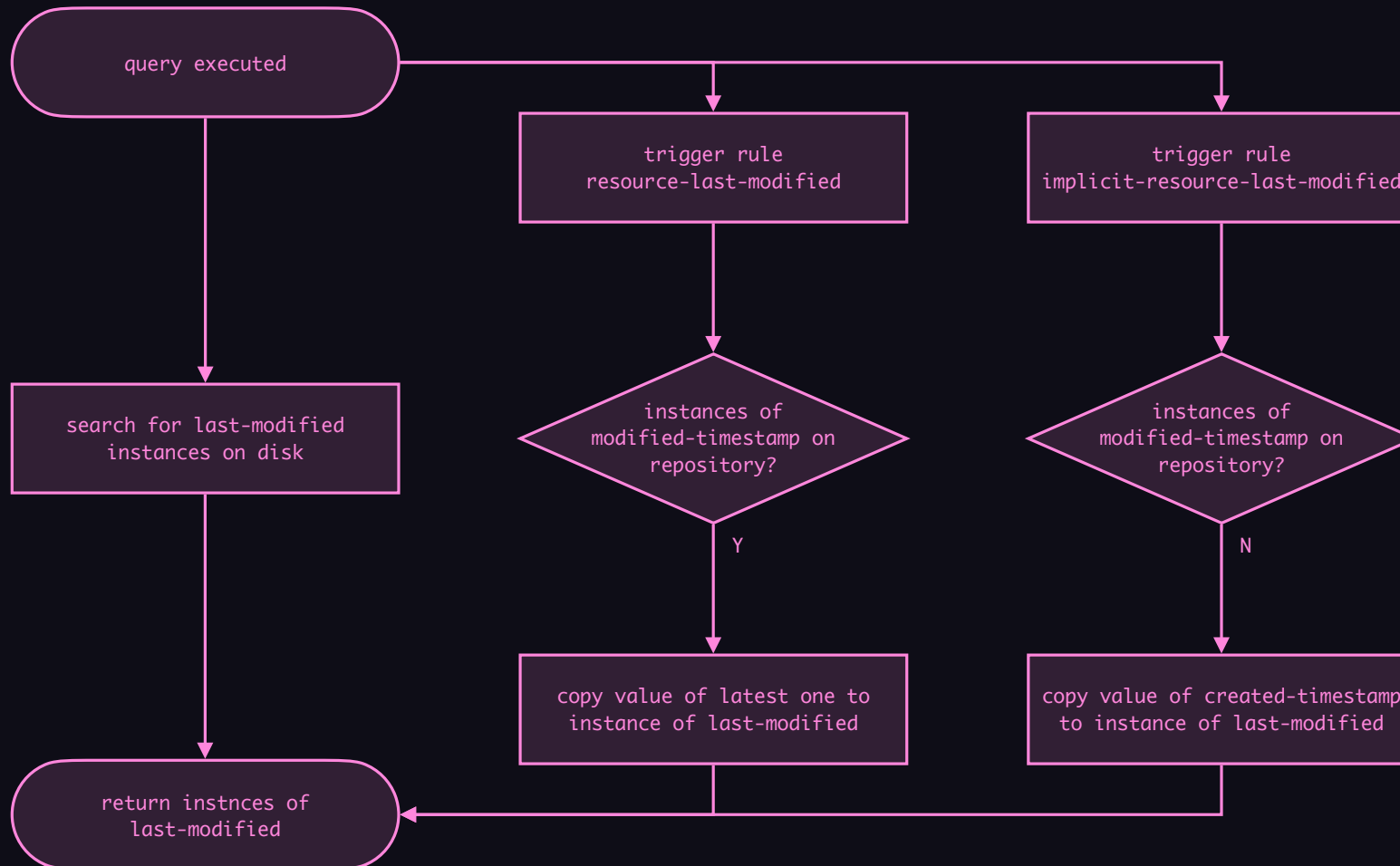
Rule branching

"List the most recent modified timestamp for each resource."



Rule branching

"List the most recent modified timestamp for each resource."



Rule chaining

"List the most recent modified timestamp for each resource."

```
match
$resource isa resource, has last-modified $t;
fetch
$resource: id;
$t;
```

Rule chaining

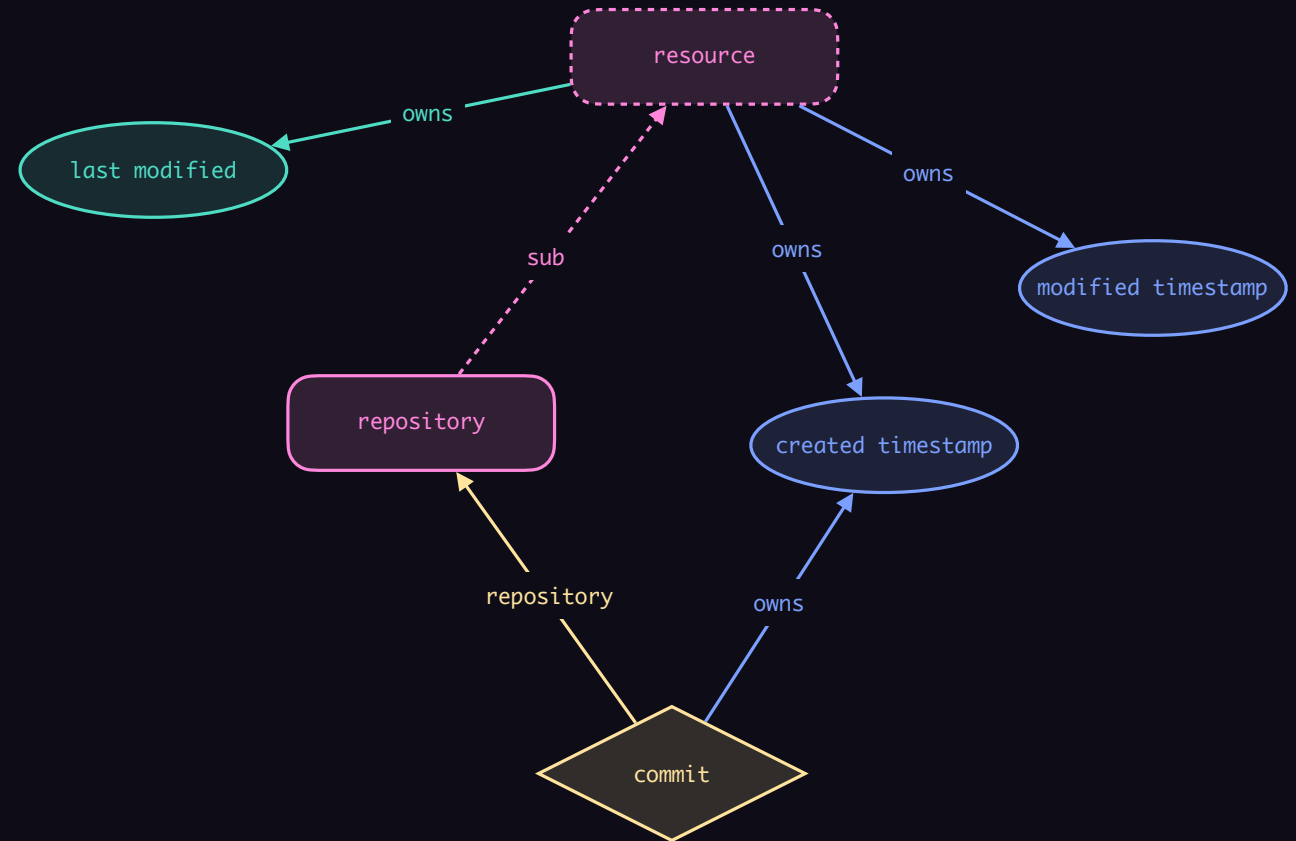
"List the most recent modified timestamp for each repository."

```
match
$repository isa repository, has last-modified $t;
fetch
$repository: id;
$t;
```


Rule chaining

"List the most recent modified timestamp for each repository."

```
match
$repository isa repository, has last-modified $t;
fetch
$repository: id;
$t;
```



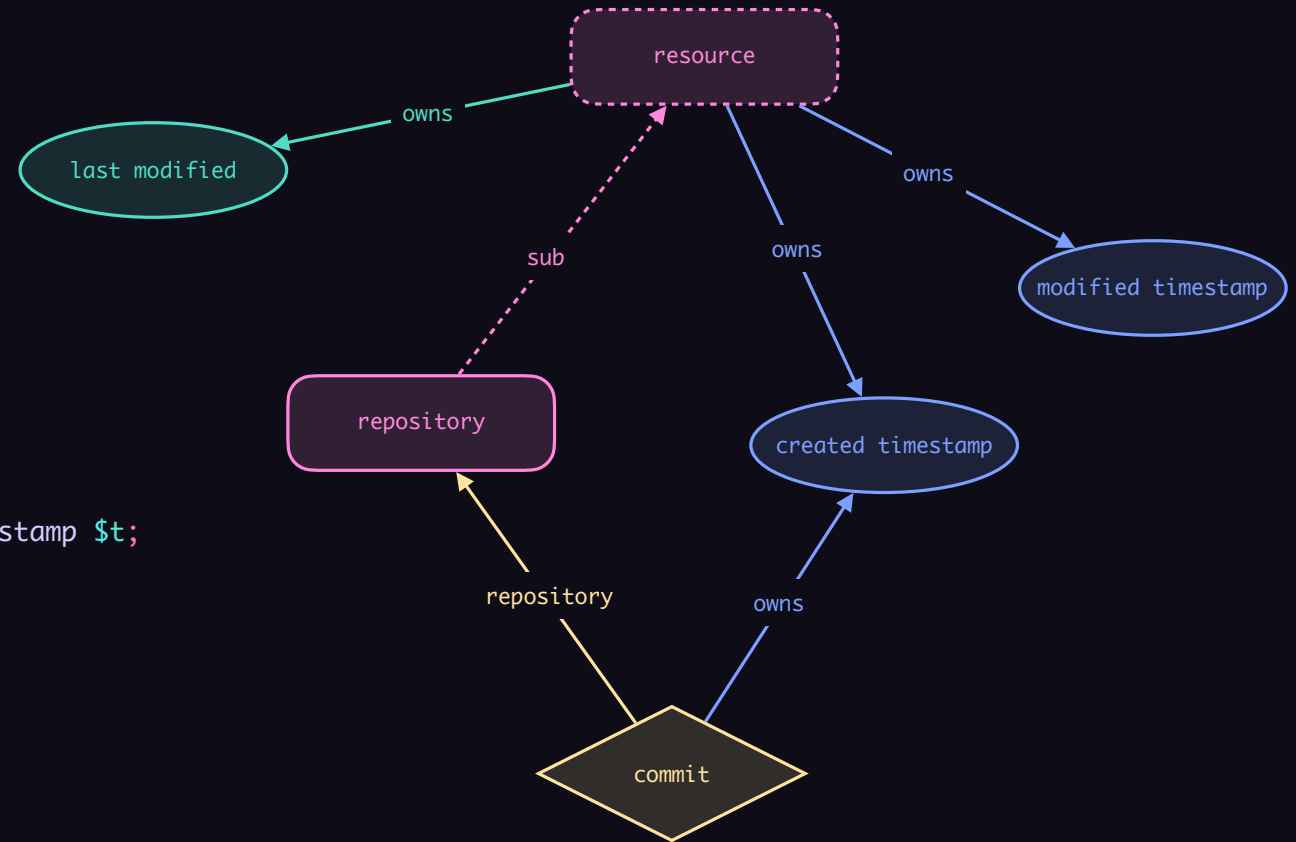
Rule chaining

"List the most recent modified timestamp for each repository."

```
match
$repository isa repository, has last-modified $t;
fetch
$repository: id;
$t;
```

define

```
rule repository-modified-timestamps:
  when {
    $repository isa repository;
    (repository: $repository) isa commit, has created-timestamp $t;
    ?new-t = $t;
  } then {
    $repository has modified-timestamp ?new-t;
  };
```



Rule chaining

"List the most recent modified timestamp for each repository."

```
match
$repository isa repository, has last-modified $t;
fetch
$repository: id;
$t;

define

rule repository-modified-timestamps:
  when {
    $repository isa repository;
    (repository: $repository) isa commit, has created-timestamp $t;
    ?new-t = $t;
  } then {
    $repository has modified-timestamp ?new-t;
  };
```

```
rule resource-last-modified:
  when {
    $resource isa resource, has modified-timestamp $t;
    not {
      $resource has modified-timestamp $t-2;
      $t-2 > $t;
    };
    ?last-t = $t;
  } then {
    $resource has last-modified ?last-t;
  };
```

Rule chaining

"List the most recent modified timestamp for each repository."

```
match
$repository isa repository, has last-modified $t;
fetch
$repository: id;
$t;
```

define

```
rule repository-modified-timestamps:
  when {
    $repository isa repository;
    (repository: $repository) isa commit, has created-timestamp $t;
    ?new-t = $t;
  } then {
    $repository has modified-timestamp ?new-t;
  };
```

```
rule resource-last-modified:
  when {
    $resource isa resource, has modified-timestamp $t;
    not {
      $resource has modified-timestamp $t-2;
      $t-2 > $t;
    };
    ?last-t = $t;
  } then {
    $resource has last-modified ?last-t;
  };
```

Condition of resource-last-modified matches
the conclusion of repository-modified-timestamps

Rule chaining

"List the most recent modified timestamp for each repository."

```
match
$repository isa repository, has last-modified $t;
fetch
$repository: id;
$t;
```

define

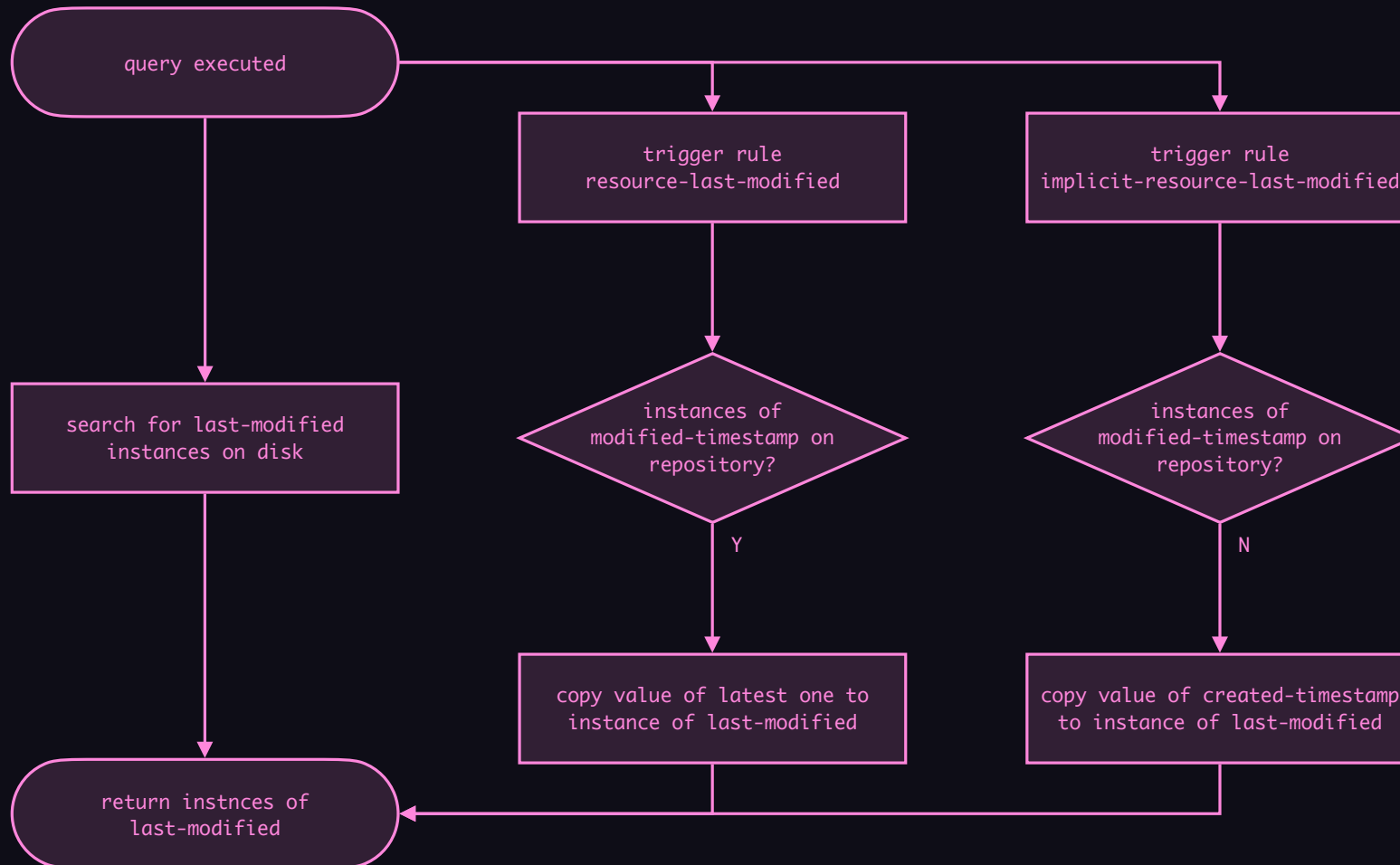
```
rule repository-modified-timestamps:
  when {
    $repository isa repository;
    (repository: $repository) isa commit, has created-timestamp $t;
    ?new-t = $t;
  } then {
    $repository has modified-timestamp ?new-t;
  };
```

```
rule implicit-resource-last-modified:
  when {
    $resource isa resource, has created-timestamp $t;
    not { $resource has modified-timestamp $t-2; };
    ?last-t = $t;
  } then {
    $resource has last-modified ?last-t;
  };
```

Condition of implicit-resource-last-modified matches
the conclusion of repository-modified-timestamps

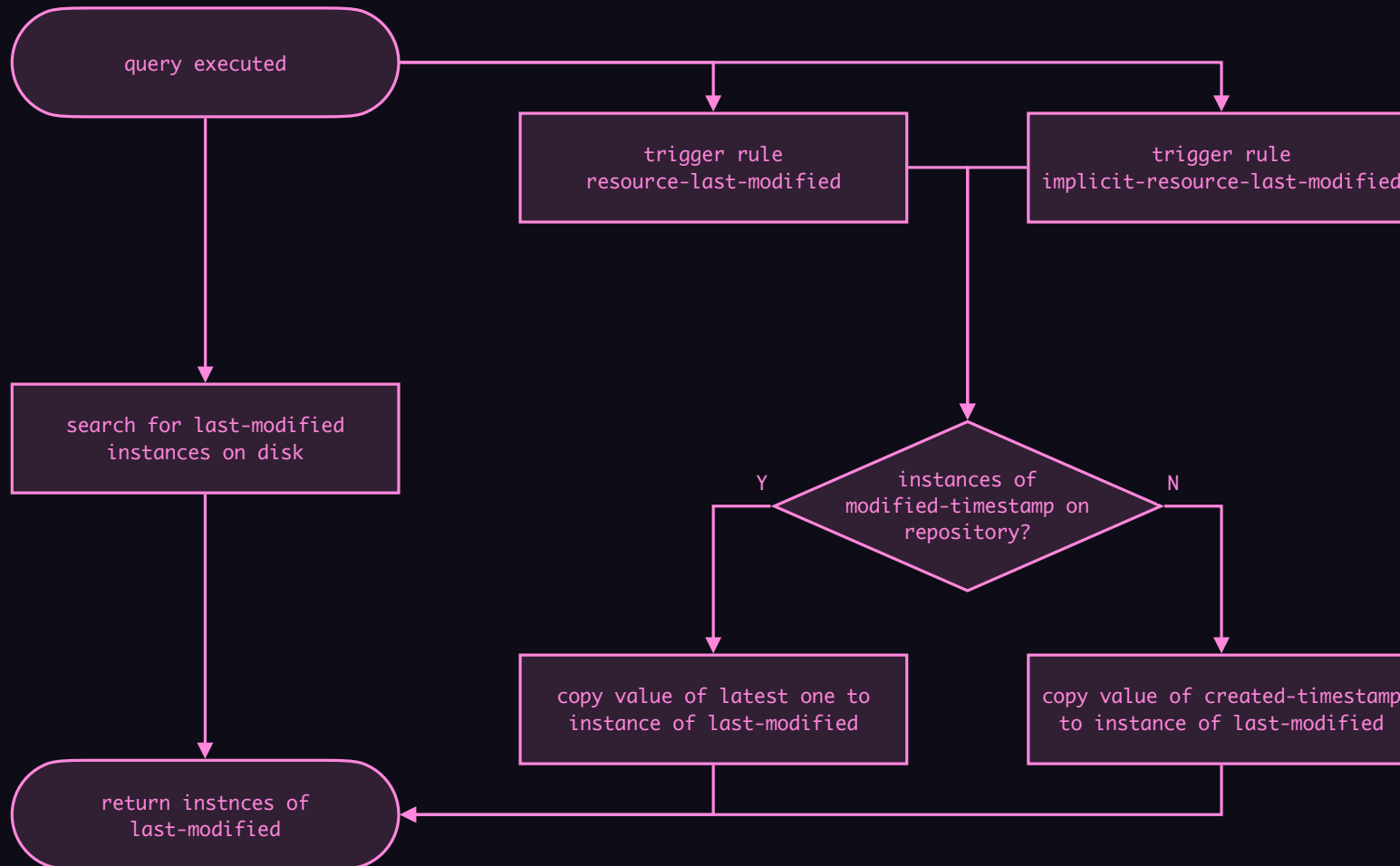
Rule chaining

"List the most recent modified timestamp for each repository."



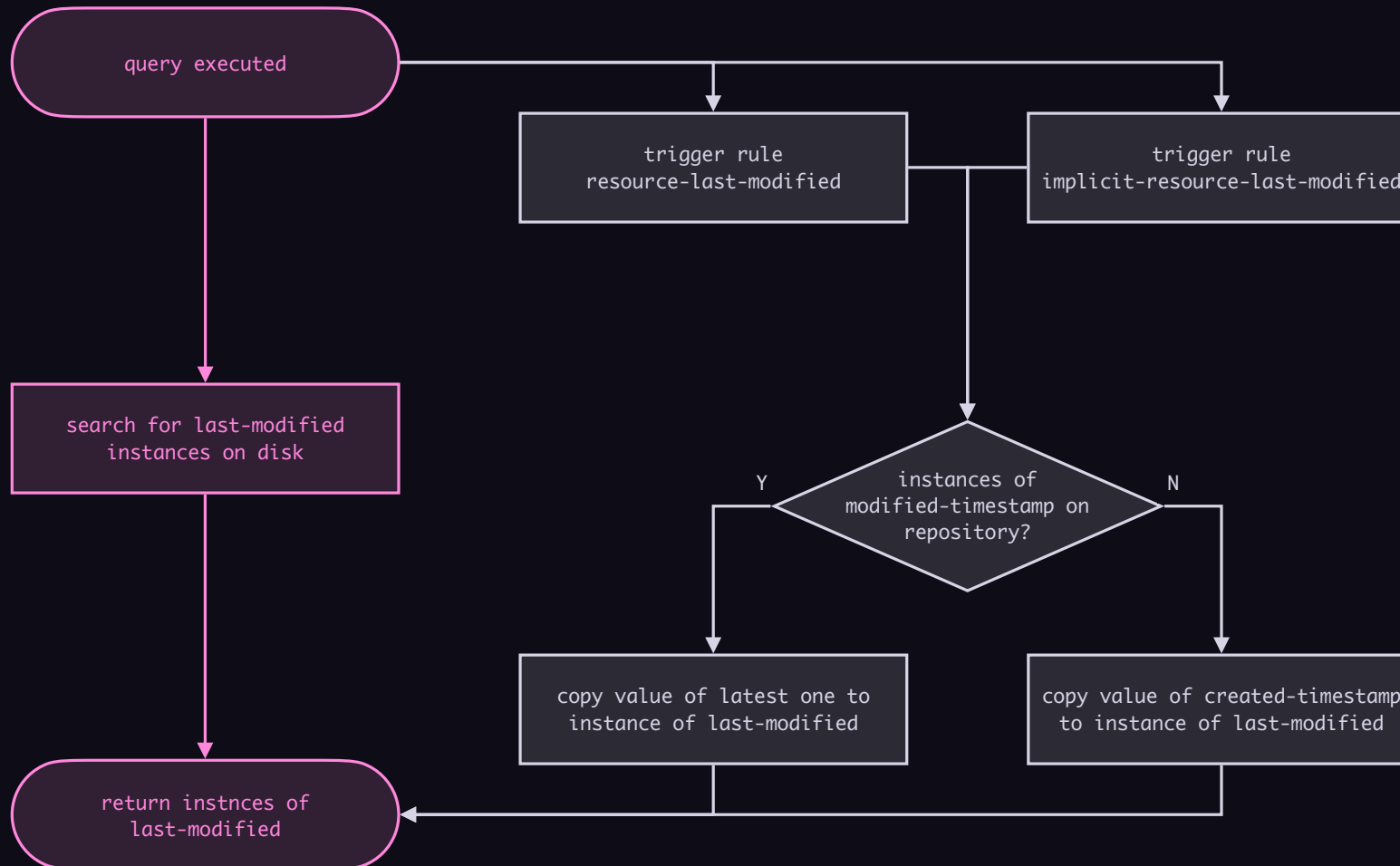
Rule chaining

"List the most recent modified timestamp for each repository."



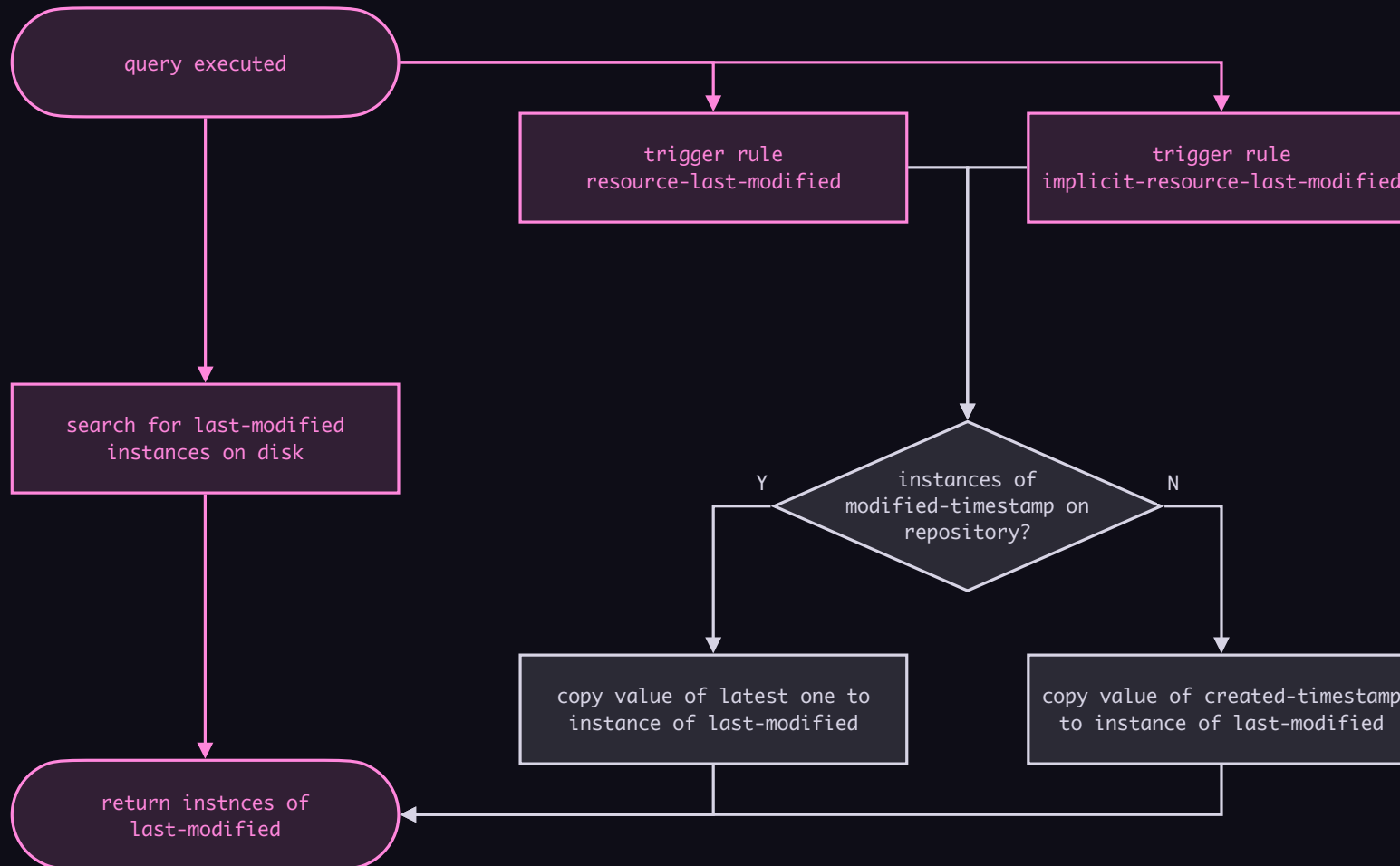
Rule chaining

"List the most recent modified timestamp for each repository."



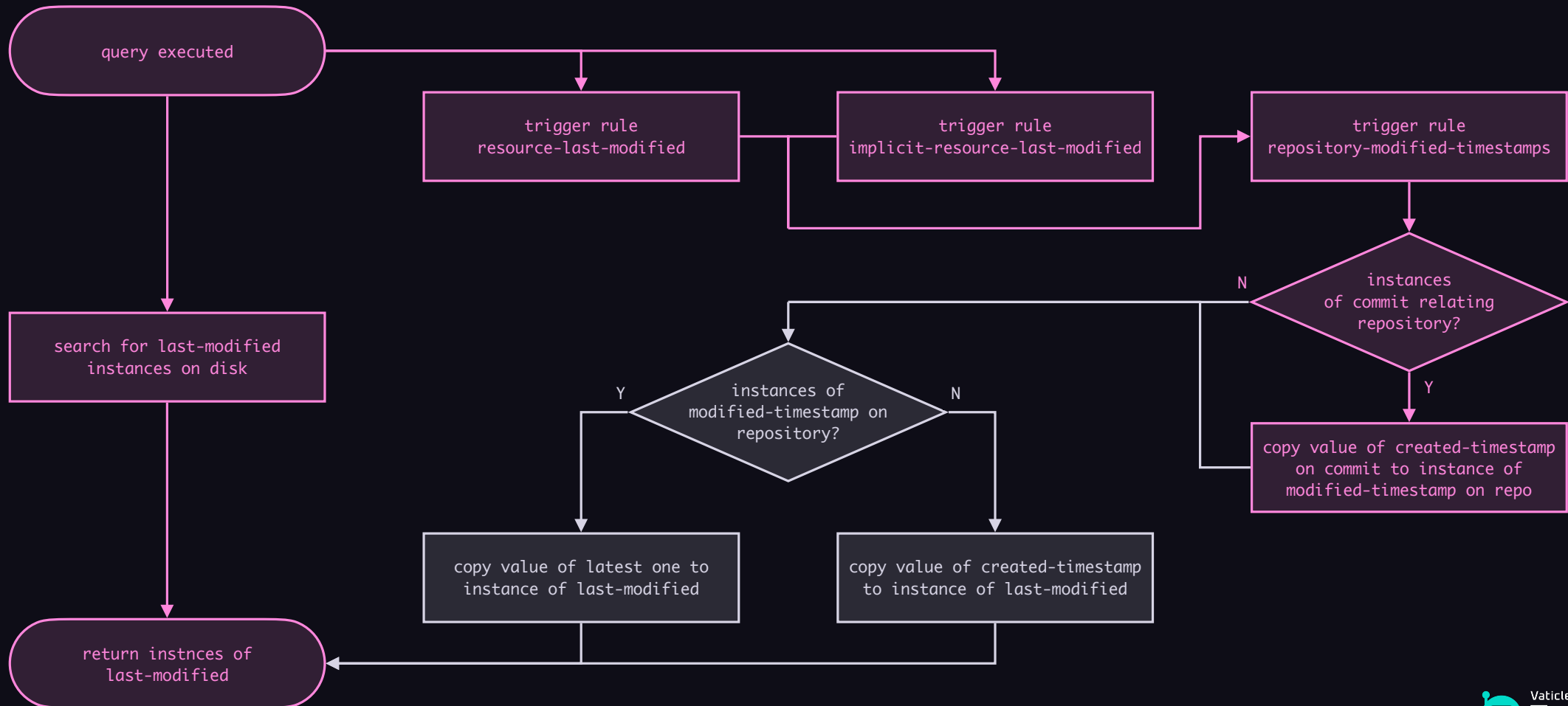
Rule chaining

"List the most recent modified timestamp for each repository."



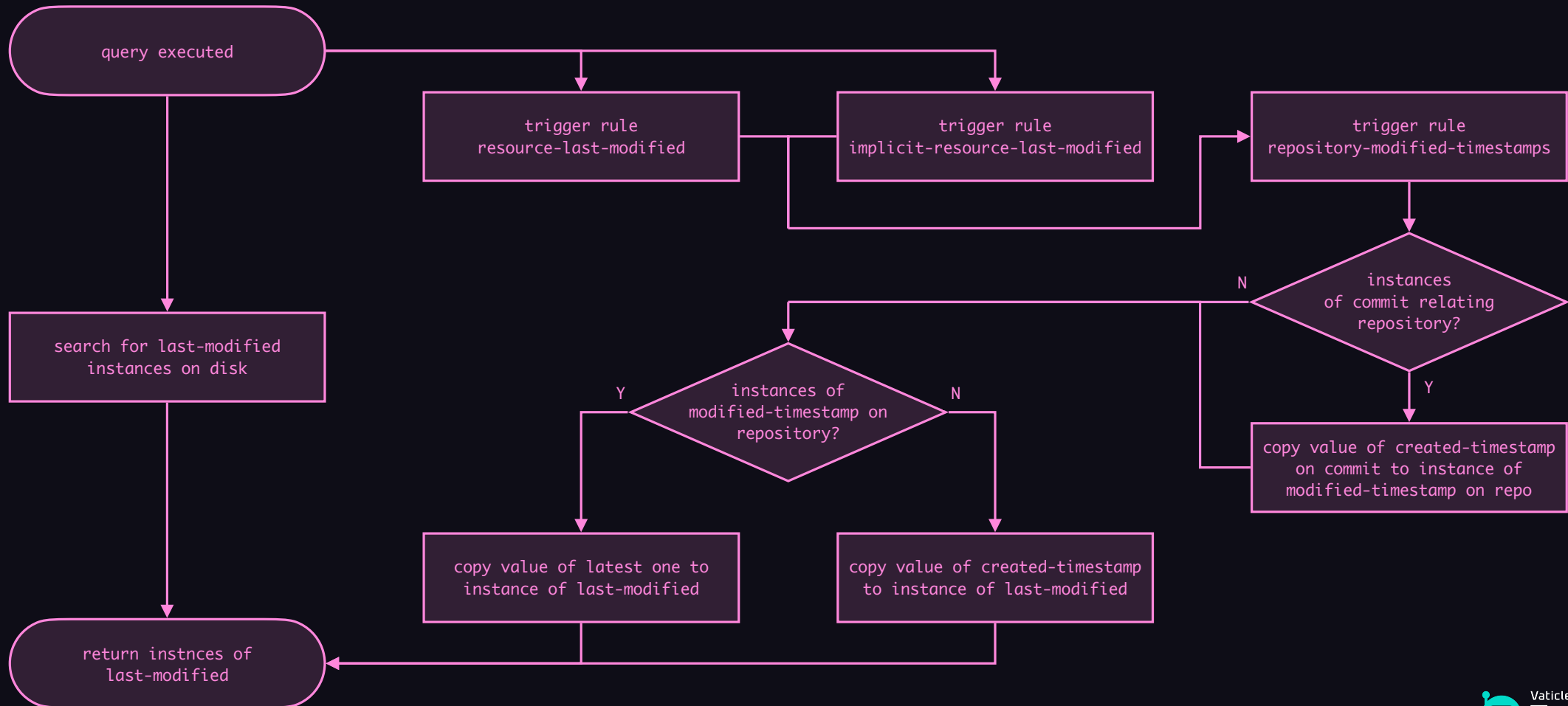
Rule chaining

"List the most recent modified timestamp for each repository."



Rule chaining

"List the most recent modified timestamp for each repository."



Rule recursion

"List the directories that the 'type theory' paper is in."

```
match
  $paper isa file, has path "/vaticle/research/papers/type-theory.tex";
  $directory isa directory, has path $path;
  (directory: $directory, directory-member: $paper) isa directory-membership;
fetch
  $path;
```

Rule recursion

"List the directories that the 'type theory' paper is in."

```
match
  $paper isa file, has path "/vaticle/research/papers/type-theory.tex";
  $directory isa directory, has path $path;
  (directory: $directory, directory-member: $paper) isa directory-membership;
fetch
  $path;
```

```
{ "path": { "value": "/vaticle/research/papers", "value_type": "string", "type": { "label": "path", "root": "attribute" } } }
```

Rule recursion

"List the directories that the 'type theory' paper is in."

```
match
  $papers isa directory, has path "/vaticle/research/papers";
  $directory isa directory, has path $path;
  (directory: $directory, directory-member: $papers) isa directory-membership;
fetch
  $path;
```

```
{ "path": { "value": "/vaticle/research", "value_type": "string", "type": { "label": "path", "root": "attribute" } } }
```

Rule recursion

"List the directories that the 'type theory' paper is in."

```
match
  $research isa directory, has path "/vaticle/research";
  $directory isa directory, has path $path;
  (directory: $directory, directory-member: $research) isa directory-membership;
fetch
  $path;
```

```
{ "path": { "value": "/vaticle", "value_type": "string", "type": { "label": "path", "root": "attribute" } } }
```

Rule recursion

"List the directories that the 'type theory' paper is in."

```
define
  implicit-directory-membership sub directory-membership;

rule transitive-directory-memberships:
  when {
    (directory: $directory-1, directory-member: $directory-2) isa directory-membership;
    (directory: $directory-2, directory-member: $x) isa! directory-membership;
  } then {
    (directory: $directory-1, directory-member: $x) isa implicit-directory-membership;
  };
```


Rule recursion

"List the directories that the 'type theory' paper is in."

```
define
  implicit-directory-membership sub directory-membership;

rule transitive-directory-memberships:
  when {
    (directory: $directory-1, directory-member: $directory-2) isa directory-membership;
    (directory: $directory-2, directory-member: $x) isa! directory-membership;
  } then {
    (directory: $directory-1, directory-member: $x) isa implicit-directory-membership;
  };
```

Condition of `implicit-directory-membership`
matches its own conclusion!

Rule recursion

"List the directories that the 'type theory' paper is in."

```
match
  $paper isa file, has path "/vaticle/research/papers/type-theory.tex";
  $directory isa directory, has path $path;
  (directory: $directory, directory-member: $paper) isa directory-membership;
fetch
  $path;
```

```
{ "path": { "value": "/vaticle/research/papers", "value_type": "string", "type": { "label": "path", "root": "attribute" } } }
```

Rule recursion

"List the directories that the 'type theory' paper is in."

```
match
  $paper isa file, has path "/vaticle/research/papers/type-theory.tex";
  $directory isa directory, has path $path;
  (directory: $directory, directory-member: $paper) isa directory-membership;
fetch
  $path;
```

```
{ "path": { "value": "/vaticle/research/papers", "value_type": "string", "type": { "label": "path", "root": "attribute" } } }
{ "path": { "value": "/vaticle/research", "value_type": "string", "type": { "label": "path", "root": "attribute" } } }
{ "path": { "value": "/vaticle", "value_type": "string", "type": { "label": "path", "root": "attribute" } } }
```

Symbolic reasoning

- Domain logic can be integrated with rules.
- Rule combinations lead to complex behaviour from simple rules.
- Many more advantages!

Summary

- Conceptual data model
- Type-theoretic language
- Strong type system
- Symbolic reasoning

Summary

- Conceptual data model → No object model mismatch
- Type-theoretic language → Declarative polymorphic queries
- Strong type system → Guaranteed semantic integrity
- Symbolic reasoning → Integrated application logic



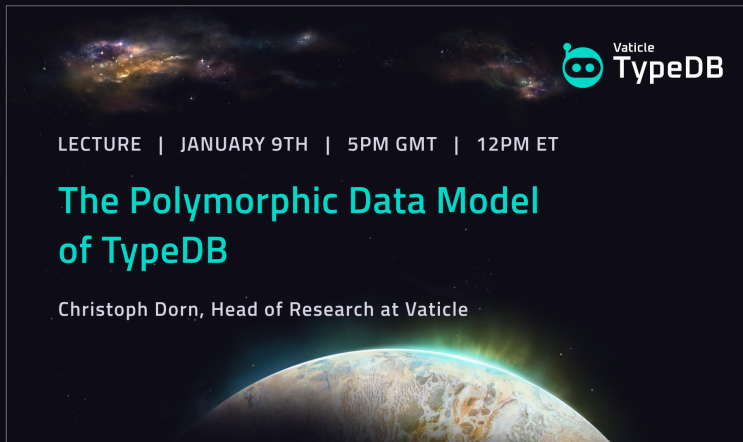
Vaticle

TypeDB

The world's first polymorphic database

There is much more to talk about, see these [upcoming lectures](#):

Tuesday, Jan 9th



Vaticle
TypeDB

LECTURE | JANUARY 9TH | 5PM GMT | 12PM ET

The Polymorphic Data Model of TypeDB

Christoph Dorn, Head of Research at Vaticle

Replay now available



TypeDB Lecture: Why We Need a Polymorphic Database

TypeDB Fundamentals Lecture Series


Why We Need a Polymorphic Database

Dr. James Whiteside
Research Engineer, Vaticle
Previously: Computational Solid-State Physicist @ University of Surrey

Watch on  YouTube



Replay now available



TypeDB Lecture: Type Theory as the Unifying Foundation for Modern Databases

TypeDB Fundamentals Lecture Series

Type Theory as the Unifying Foundation for Modern Databases

Dr. Christoph Dorn
Head of Research, Vaticle
Previously: Theoretical Computer Scientist in Category Theory @ Oxford University

Watch on  YouTube



Register or watch at TypeDB.com/lectures



Q & A

More TypeDB Resources



TypeDB Learning Center
typedb.com/learn



Download TypeDB
typedb.com/deploy



TypeDB Cloud
cloud.typedb.com



Vaticle

TypeDB

Thank you!

Join us at typedb.com/discord