User(user_id, name, email, role)

Primary key: user_id

Foreign key: NONE

Data types:

- user_id: INTEGER
- name: VARCHAR(100)
- email: VARCHAR(100) UNIQUE
- role: VARCHAR(100)

Functional dependencies:

- user_id->name, email, role
- email->user_id, name, role

---

Environment(env_id, name, is_active, file_id)

Primary key: env_id

Foreign key: file_id->Config_File.file_id

Data types:

- env_id: INTEGER
- name: VARCHAR(100)
- is_active: BOOLEAN

Functional dependencies:

- env_id->name, is_active, file_id

---

Tool_Category(category_id, category_name)

Primary key: category_id

Foreign key: NONE

Data types:

- category_id: INTEGER
- category_name: VARCHAR(100)

Functional dependencies:

- category_id->category_name

---

Source_Ctrl(tool_id, name, type, version, category_id)

Primary key: tool_id

Foreign key: category_id->Tool_Category.category_id

Data types:

- tool_id: INTEGER
- name: VARCHAR(100)
- type: VARCHAR(100)
- version: VARCHAR(50)
- category_id: INTEGER

Functional dependencies:

- tool_id->name, type, version, category_id

---

Testing(tool_id, name, type, version, category_id)

Primary key: tool_id

Foreign key: category_id->Tool_Category.category_id

Data types:

- tool_id: INTEGER
- name: VARCHAR(100)
- type: VARCHAR(100)
- version: VARCHAR(50)
- category_id: INTEGER

Functional dependencies:

- tool_id->name, type, version, category_id

---

Config_File(file_id, name, hash, user_id)

Primary key: file_id

Foreign key: user_id->User.user_id

Data types:

- file_id: INTEGER
- name: VARCHAR(100)
- hash: VARCHAR(100)
- user_id: INTEGER

Functional dependencies:

- file_id->name, hash, user_id

---

Service(service_id, name, is_active)

Primary key: service_id

Foreign key:

Data types:

- service_id: INTEGER
- name: VARCHAR(100)
- is_active: BOOLEAN

Functional dependencies:

- service_id->name, is_active

---

Pipeline(pipe_id, name, service_id)

Primary key: pipe_id

Foreign key: service_id->Service.service_id

Data types:

- pipe_id: INTEGER
- name: VARCHAR(100)
- service_id: INTEGER

Functional dependencies:

- pipe_id->name, service_id

---

Pipeline_Step(pipe_id, name, step_id)

Primary key: pipe_id, step_id

Foreign key: pipe_id->Pipeline.pipe_id

Data types:

- pipe_id: INTEGER
- name: VARCHAR(100)
- step_id: INTEGER

Functional dependencies:

- pipe_id, step_id -> name

---

Deployment(deploy_id, status, version, timestamp, user_id, env_id, service_id)

Primary key: deploy_id

Foreign key:

- user_id->User.user_id
- env_id->Environment.env_id
- service_id->Service.service_id

Data types:

- deploy_id: INTEGER
- status: VARCHAR(100)
- version: VARCHAR(50)
- timestamp: TIMESTAMPTZ
- user_id: INTEGER
- env_id: INTEGER
- service_id: INTEGER

Functional dependencies:

- deploy_id->status, version, timestamp, user_id, env_id, service_id

---

Log(log_id, type, timestamp, deploy_id)

Primary key: log_id

Foreign key: deploy_id->Deployment.deploy_id

Data types:

- log_id: INTEGER
- type: VARCHAR(100)
- timestamp: TIMESTAMPTZ
- deploy_id: INTEGER

Functional dependencies:

- log_id->type, timestamp, deploy_id

---

Uses(user_id, tool_id, env_id)

Primary key: user_id, category_id, env_id

Foreign key:

- user_id->User.user_id

- category_id->Tool_Category.category_id
- env_id->Environment.env_id

Data types:

- user_id: INTEGER
- category_id: INTEGER
- env_id: INTEGER

Functional dependencies: NONE NONTRIVIAL

---

Accesses(<u>user_id, service_id</u>, permissions)

Primary key: user_id, service_id

Foreign key:

- user_id->User.user_id
- service_id->Service.service_id

Data types:

- user_id: INTEGER
- service_id: INTEGER
- permissions: VARCHAR(100)

Functional dependencies:

- user_id, service_id -> permissions

My ER diagram had a few flaws with it. For one, I did not properly implement the subclasses for Tool, and the attributes were the same for the superclass and its subclasses. I also got a few of the cardinalities wrong upon further inspection. I was able to decompose the “has,” “what,” “who,” “where,” “generates,” and “configures” relationships as they were all either 1-1 or 1-M. “accesses,” and “uses” were made into their own tables as one relationship was M-N, while the other was a three-way ternary relationship. There were no examples of non-BCNF relation schemas, since I gave most tables unique ID identifiers for their primary keys. I also changed the name of the “Tool” superclass to “Tool_Category” to be more descriptive. I think typical queries for the database will be centered around either the “User” or “Deployment” entities, as they are most instrumental to the database. Overall, I am very happy with how it turned out.