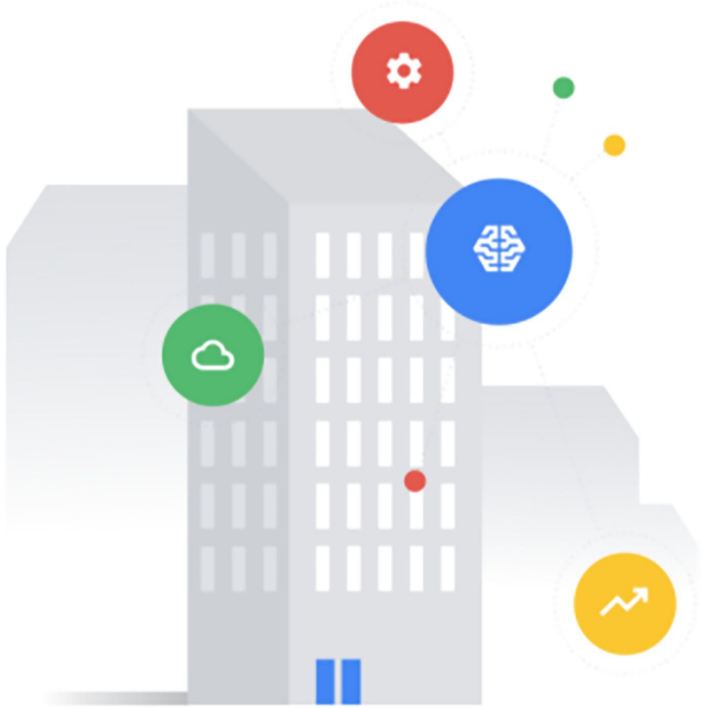
Google

Module 1 | **Lesson 8**

Digital Buildings Ontology (DBO)



Before you get started

This learning module has interactive features and activities that enable a self-guided learning experience. To help you get started, here are two tips for viewing and navigating through the content.

1 View this content outside of GitHub.

- For the best learning experience, you're encouraged to download a copy so links and other interactive features will be enabled.
- To download a copy of this lesson, click **Download** in the top-right corner of this content block.
- After downloading, open the file in your preferred PDF reader application.

2 Navigate by clicking the buttons and links.

- For the best learning experience, using your keyboard or mouse wheel to navigate is discouraged. However, this is your only option if you're viewing from GitHub.
- If you're viewing this content outside of GitHub:
  - Click the **Back** or **Next** buttons to go backward or forward in the deck. Moving forward, you'll find them in the bottom corners of every slide.
  - Click blue text to go to another slide in this deck or open a new page in your browser.

Ready to get started? Let's go!

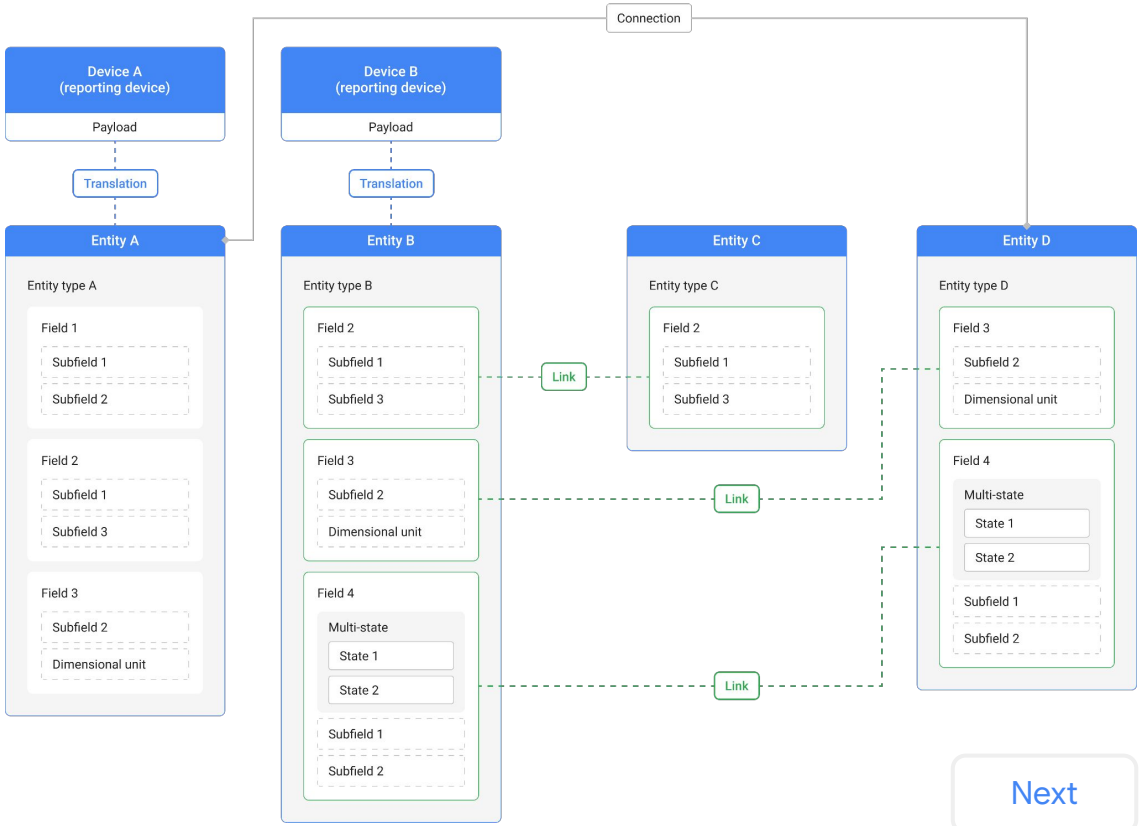
Conceptual model revisited

Here's another look at the DBO conceptual model from Lesson 2.

In this lesson, you'll explore one modeling concept from the abstract model. Remember, the following modeling concepts are used to describe the relationships that can occur between entities:

- Mappings
  - Translations
  - Links
- Connections

Do you see these concepts in the diagram?



Back

Next

Entities and their GUIDs

We identify entities using a **globally unique identifier (GUID)** to group their data and link or connect them to other entities.

Connections are encoded using the **building config format**. The example below shows a connection in the new building config format and highlights the GUIDs of the entity and its connections. GUIDs can be created in advance using any GUID generator (like UUID).

If working with the old building config format, the Digital Building Project's GUID Generator can convert it into the new format shown below and add the GUIDs.

Examples in this lesson use a **human-readable** version of the new format and reference an entity by its "code." The example below shows the same connection and highlights the differences. We'd reference this entity by its code **VAV-32**.

This format is only meant to support readability in this lesson, so it shouldn't be used in your actual work. Outside of this lesson, always use the correct building config format with an actual GUID.

Building config format

```
5e0bc6fa-e85f-4149-a3df-c5bcc0c3d9b9:
  code: VAV-32
  type: HVAC/VAV
  connections:
    d8de8611-d327-4957-9f29-fb5cd6f58905: CONTAINS
    c6358fa8-8630-444f-bb15-9e4c38d21271: FEEDS
```

Human-readable format 👓

```
replace-with-guid-of-vav-32:
  code: VAV-32
  type: HVAC/VAV
  connections:
    replace-with-guid-of-bldg-1: CONTAINS
    replace-with-guid-of-ahu-1: FEEDS
```

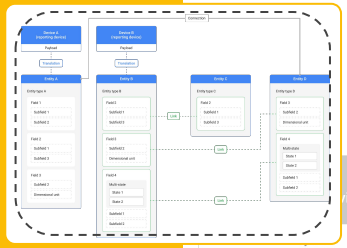
Back

Note: For the remainder of this lesson, the examples marked with glasses 👓 are using the human-readable format of the building config. We'll also refer to entities by their code to further support readability. Outside of this lesson, always use the correct building config format with an actual GUID in your work!

Next

Lesson 8

Connections

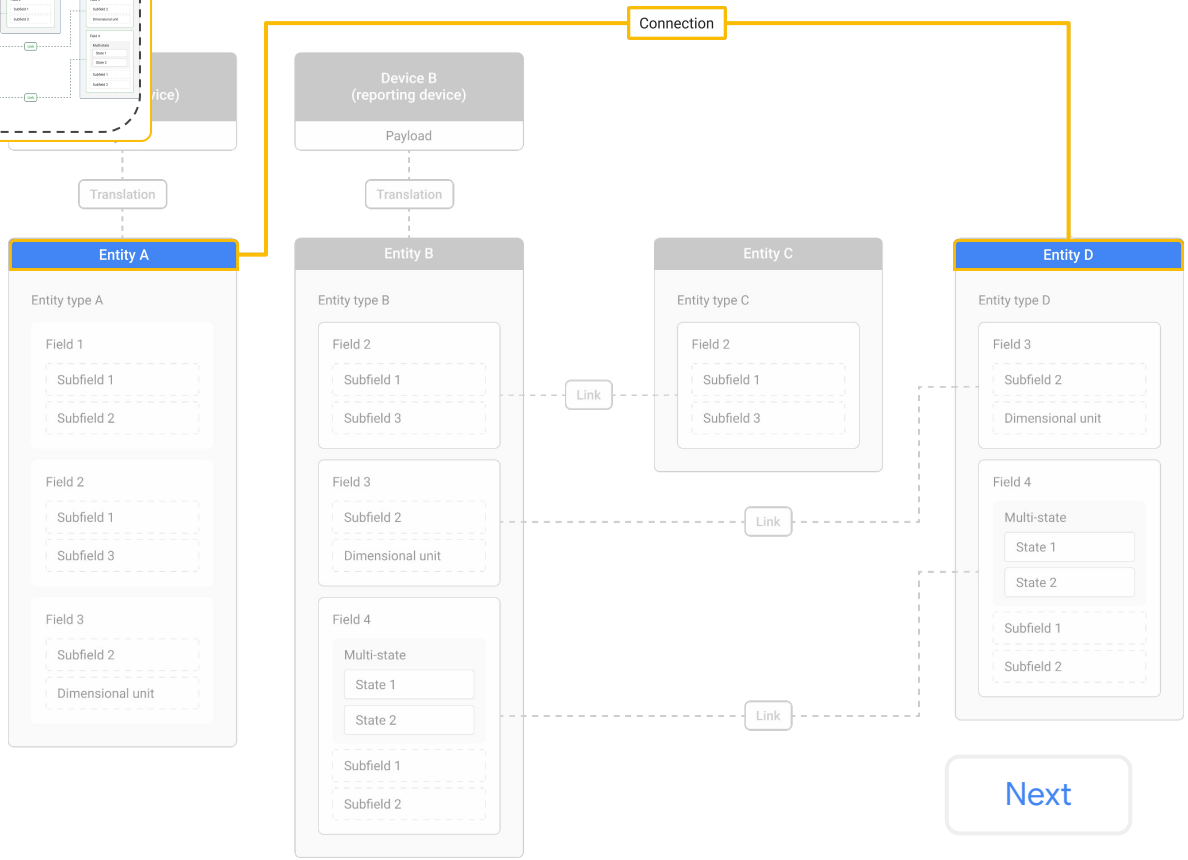


What you'll learn about:

- Connections
- Connection definitions
- Source and target entities
- Connection construction syntax

By the end of this lesson, you'll be able to:

- Describe the concept of a connection.
- Identify a connection in source code.
- Recognize the different connection definitions.
- Understand the relationship between source and target entities.
- Construct a valid connection.



Back

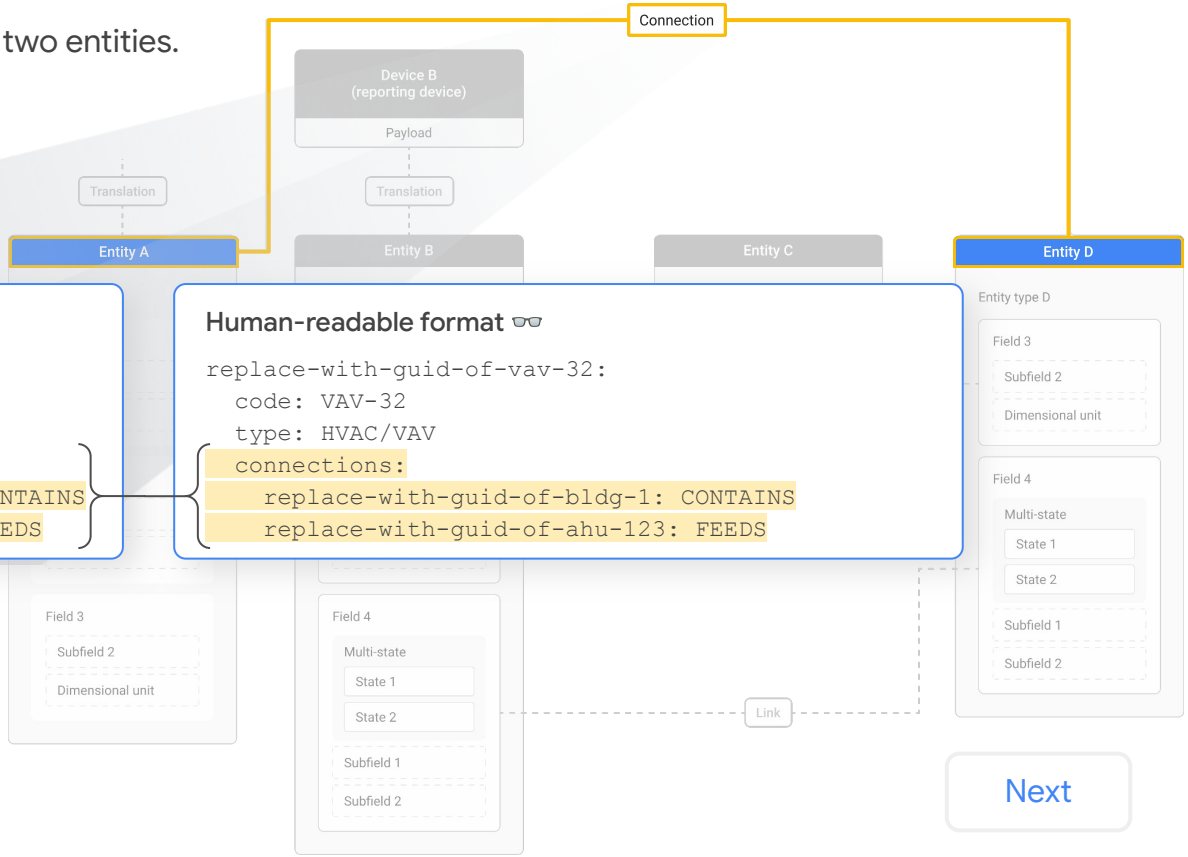
Next

Google

What’s a connection?

A connection is a named relationship between two entities.

Each connection has a specific definition that describes the relationship.



Connection

```
5e0bc6fa-e85f-4149-a3df-c5bcc0c3d9b9:
  code: VAV-32
  type: HVAC/VAV
  connections:
    d8de8611-d327-4957-9f29-fb5cd6f58905: CONTAINS
    c6358fa8-8630-444f-bb15-9e4c38d21271: FEEDS
```

Human-readable format 👓

```
replace-with-guid-of-vav-32:
  code: VAV-32
  type: HVAC/VAV
  connections:
    replace-with-guid-of-bldg-1: CONTAINS
    replace-with-guid-of-ahu-123: FEEDS
```

Back

Next

Connection definitions

There are five different types of connection definitions.

Each type of connection can be used to define the various system and spatial relationships that occur between two entities.

CONTAINS

This connection is used to show the source entity physically encapsulates at least part of the target entity.

CONTROLS

This connection is used to show the source entity determines or affects the internal state or behavior of the target entity.

FEEDS

This connection is used to show the source entity provides some media (e.g., water or air) to the target entity.

HAS_PART

This connection is used to show the source entity has a component or part that's defined by the target entity.

HAS_RANGE

This connection is used to show the source entity has a coverage or detection range defined by the target entity.

Back

Note: Connection definitions are always defined in the global namespace. See connections.yaml in the Digital Buildings Project GitHub repo for a list of available connection definitions.

Next

Source and target entities

A connection is a directional relationship going from a source to a target.

A **source entity** is the entity that connects to another entity.

A **target entity** is the entity that is being connected.

Why does this matter?

It's important to properly identify the source entity and the target entity because valid connection construction follows a specific syntax. As shown in the example to the right, identifying source and target entities should be fairly intuitive in most cases.

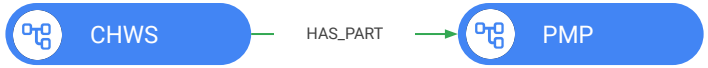
Connections are always defined on the target entity. If a connection is mistakenly defined on the source entity, then it'll run into a validation deadlock. When a validation deadlock occurs, the source entity is invalidated and the target entity is deleted.

Example

A building contains a floor. The building is the source entity. The floor is the target entity. The defined connection is `contains`.

An air handling unit (AHU) feeds a variable air volume (VAV). The AHU is the source entity. The VAV is the target entity. The defined connection is `feeds`.

A chilled water system (CHWS) has a pump (PMP). The CHWS is the source entity. The PMP is the target entity. The defined connection is `has_part`.



Back

Note: Connections are always defined on the target entity. If not, then a validation deadlock can occur. When a validation deadlock occurs, the source entity is invalidated and the target entity is deleted. Follow the appropriate construction syntax to avoid this!

Next

Connection construction syntax

Connections are always defined on the target entity.

A connection is defined with the **`connections`** block.
A target entity will use the block to define its source entity. Then, the connection is defined with one of the five connection definitions.

First, you'll see the GUID of the target entity.

Connection (Click **Next** to view the human-readable format.)

```
4q2ka6kv-2s7m-2226-n2kp-q1nuk1h1k4j5
  code: LF-123
  type: LIGHTING/LIGHTING_FIXTURE
  connections:
    d8de8611-d327-4957-9f29-fb5cd6f58905 CONTAINS
    y2215yj0-1322-122n-eh23-2q2j43p53261 HAS_PART
```

In the **`connections`** block, you'll see the GUIDs of the source entities...

...along with the connection definition.

Example (Click **Next** to view the human-readable format.)

The entity coded **`VAV-32`** has a defined connection:

```
5e0bc6fa-e85f-4149-a3df-c5bcc0c3d9b9 :
  code: VAV-32
  type: HVAC/VAV
  connections:
    d8de8611-d327-4957-9f29-fb5cd6f58905: CONTAINS
    c6358fa8-8630-444f-bb15-9e4c38d21271 : FEEDS
```

- The target entity is **`5e0bc6fa-e85f-4149-a3df-c5bcc0c3d9b9`**, which is the GUID for a variable air volume system coded **`VAV-32`**.
- The source entity is **`c6358fa8-8630-444f-bb15-9e4c38d21271`**, which is the GUID for an air handling unit that's been coded **`AHU-123`**.
- The defined connection is **`FEEDS`**, which means the source entity provides media to the target entity.

Note the syntax. The target entity coded **`VAV-32`** has defined its connection to the source entity coded **`AHU-123`**. This means **`AHU-123`** feeds media to **`VAV-32`**. Put simply, an air handling unit feeds air to a variable air volume system.

It's counterintuitive to switch **`AHU-123`** and **`VAV-32`** in the model. The reason the syntax is defined this way is to allow for the deletion of an entity (along with its connections) without breaking the source. This can be useful in editing of building config files.

Back

Next

Connection construction syntax (continued)

Connections are always defined on the target entity.

A connection is defined with the **`connections`** block.
A target entity will use the block to define its source entity. Then, the connection is defined with one of the five connection definitions.

First, you'll see the GUID of the target entity.

Connection 👓

```
replace-with-guid-of-lf-123:
  code: LF-123
  type: LIGHTING/LIGHTING_FIXTURE
  connections:
    replace-with-guid-of-bldg-1: CONTAINS
    replace-with-guid-of-lcg-123: HAS_PART
```

In the **`connections`** block, you'll see the GUIDs of the source entities...

...along with the connection definition.

Example 👓

The entity coded **`VAV-32`** has a defined connection:

```
replace-with-guid-of-vav-32 : #NOTE Target entity.
  code: VAV-32
  type: HVAC/VAV
  connections:
    replace-with-guid-of-bldg-1: CONTAINS
    replace-with-guid-of-ahu-123 : FEEDS #NOTE Source
entity.
```

- The target entity is **`VAV-32`**, which is a variable air volume system.
- The source entity is **`AHU-123`**, which is an air handling unit.
- The defined connection is **`FEEDS`**, which means the source entity provides media to the target entity.

Note the syntax. **`VAV-32`**, the target entity, has defined its connection to **`AHU-123`**, the source entity. This means **`AHU-123`** feeds media to **`VAV-32`**. Put simply, an air handling unit feeds air to a variable air volume system.

It's counterintuitive to switch **`AHU-123`** and **`VAV-32`** in the model. The reason the syntax is defined this way is to allow for the deletion of an entity (along with its connections) without breaking the source. This can be useful in editing of building config files.

Back

Note: Remember, this example is using the human-readable format of the building config (as indicated by the glasses 👓). In your work outside of this lesson, connections should always be encoded in the correct building config format using an actual GUID.

Next

Lesson 8

Knowledge check

Let's take a moment to reflect on what you've learned so far.

- The next slides will have questions about the concepts that were introduced in this lesson.
- Review each question and select the correct response.

If there are more than two answer options, you won't be able to move forward until the correct answer is selected.

Back

*Click **Next** when you're ready to begin.*

Next

Google

Knowledge check 1

A connection is a named relationship between two _____.

Fill in the blank.

Select the best answer from the options listed below.

- entities
- mappings
- namespaces
- entity types

Back

Next

Knowledge check 1

A connection is a named relationship between two _____.

Fill in the blank.

Select the best answer from the options listed below.

- entities
- mappings
- namespaces
- entity types

That's right! 🎉

Connections occur between two **entities**.

To define a connection, you'll use the `connections` block and apply one of the following definitions:

- `CONTAINS`
- `CONTROLS`
- `FEEDS`
- `HAS_PART`
- `HAS_RANGE`

Back

Next

Knowledge check 1

A connection is a named relationship between two _____.

Fill in the blank.

Select the best answer from the options listed below.

- entities
- mappings
- namespaces
- entity types

Hmm, that's not right! 🤔

Try again

Back

Next

Knowledge check 1

A connection is a named relationship between two _____.

Fill in the blank.

Select the best answer from the options listed below.

- entities
- mappings
- namespaces
- entity types

Hmm, that's not right! 🤔

Try again

Back

Next

Knowledge check 1

A connection is a named relationship between two _____.

Fill in the blank.

Select the best answer from the options listed below.

- entities
- mappings
- namespaces
- entity types

Hmm, that's not right! 🤔

Try again

Back

Next

Knowledge check 2

Connections must be defined on a specific entity to avoid a validation deadlock.

On which entity should a connection be defined?

Select the best answer from the options listed below.

On the source entity

On the target entity

Back

Next

Knowledge check 2

Connections must be defined on a specific entity to avoid a validation deadlock.

On which entity should a connection be defined?

Select the best answer from the options listed below.

- On the source entity
- On the target entity

Hmm, that's not right! 🤔

Connections are always defined **on the target entity**.

Target entity

Source entities

Connection definitions

Connection 👓

```
replace-with-guid-of-lf-123 :
  code: LF-123
  type: LIGHTING/LIGHTING_FIXTURE
  connections:
    replace-with-guid-of-bldg-1 : CONTAINS
    replace-with-guid-of-lcg-123 : HAS_PART
```

If a connection is defined on the source entity, then a validation deadlock occurs. This is problematic because the source entity will be invalidated and the target entity will be deleted.

Back

Note: Remember, this example is using the human-readable format of the building config (as indicated by the glasses 👓). In your work outside of this lesson, connections should always be encoded in the correct building config format using an actual GUID.

Next

Knowledge check 2

Connections must be defined on a specific entity to avoid a validation deadlock.

On which entity should a connection be defined?

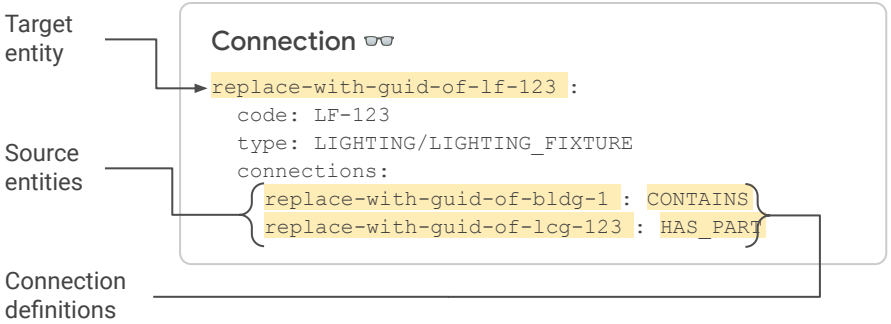
Select the best answer from the options listed below.

On the source entity

On the target entity

That's right! 🎉

Connections are always defined **on the target entity**.



If a connection is defined on the source entity, then a validation deadlock occurs. This is problematic because the source entity will be invalidated and the target entity will be deleted.

Back

Note: Remember, this example is using the human-readable format of the building config (as indicated by the glasses 👓). In your work outside of this lesson, connections should always be encoded in the correct building config format using an actual GUID.

Next

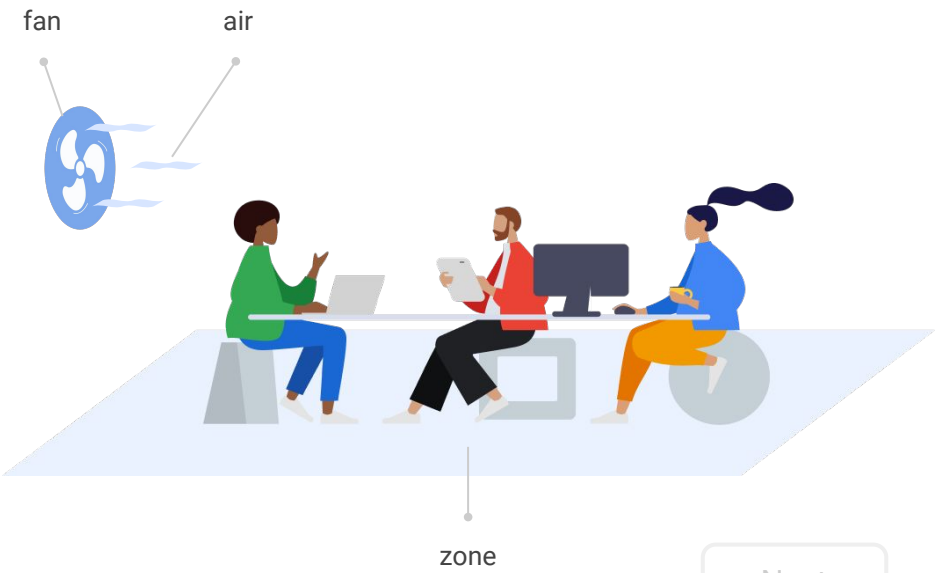
Knowledge check 3

Let's say you have a fan that delivers air to a single zone. You'd like to model the **FAN** and the **ZONE**, and then define a connection between them.

Which is the most appropriate connection definition?

Select the best answer from the options listed below.

- **FAN** is the target, **ZONE** is the source, **FEEDS** is the connection
- **FAN** is the source, **ZONE** is the target, **FEEDS** is the connection
- **FAN** is the target, **ZONE** is the source, **CONTAINS** is the connection
- **FAN** is the source, **ZONE** is the target, **CONTAINS** is the connection



Back

Next

Knowledge check 3

Let's say you have a fan that delivers air to a single zone. You'd like to model the **FAN** and the **ZONE**, and then define a connection between them.

Which is the most appropriate connection definition?

Select the best answer from the options listed below.

- **FAN** is the target, **ZONE** is the source, **FEEDS** is the connection
- FAN is the source, ZONE is the target, FEEDS is the connection
- FAN is the target, ZONE is the source, CONTAINS is the connection
- FAN is the source, ZONE is the target, CONTAINS is the connection

Close... but not quite right! 🤔

In this scenario, it doesn't make sense for the **fan** to be the target entity since it's delivering air to the **zone**.

Try again

Back

Next

Knowledge check 3

Let's say you have a fan that delivers air to a single zone. You'd like to model the **FAN** and the **ZONE**, and then define a connection between them.

Which is the most appropriate connection definition?

Select the best answer from the options listed below.

- FAN is the target, ZONE is the source, FEEDS is the connection
- **FAN** is the source, **ZONE** is the target, **FEEDS** is the connection
- FAN is the target, ZONE is the source, CONTAINS is the connection
- FAN is the source, ZONE is the target, CONTAINS is the connection

That's right! 🎉

The **FAN** is the source entity. The **ZONE** is the target entity. The connection type is **FEEDS**.

Connection 👓

```
replace-with-guid-of-zone-123 :
  code: ZONE-123
  type: HVAC/ZONE
  connections:
    replace-with-guid-of-bldg-1: CONTAINS
    replace-with-guid-of-fan-123 : FEEDS
```

Back

Note: Remember, this example is using the human-readable format of the building config (as indicated by the glasses 👓). In your work outside of this lesson, connections should always be encoded in the correct building config format using an actual GUID.

Next

Knowledge check 3

Let's say you have a fan that delivers air to a single zone. You'd like to model the **FAN** and the **ZONE**, and then define a connection between them.

Which is the most appropriate connection definition?

Select the best answer from the options listed below.

- FAN is the target, ZONE is the source, FEEDS is the connection
- FAN is the source, ZONE is the target, FEEDS is the connection
- **FAN** is the target, **ZONE** is the source, **CONTAINS** is the connection
- FAN is the source, ZONE is the target, CONTAINS is the connection

Close... but not quite right! 🤔

In this scenario, it doesn't make sense for the **fan** to be the target entity since it's delivering air to the **zone**. The **zone** might not contain the **fan** either.

Try again

Back

Next

Knowledge check 3

Let's say you have a fan that delivers air to a single zone. You'd like to model the **FAN** and the **ZONE**, and then define a connection between them.

Which is the most appropriate connection definition?

Select the best answer from the options listed below.

- FAN is the target, ZONE is the source, FEEDS is the connection
- FAN is the source, ZONE is the target, FEEDS is the connection
- FAN is the target, ZONE is the source, CONTAINS is the connection
- **FAN** is the source, **ZONE** is the target, **CONTAINS** is the connection

Close... but not quite right! 🤔

In this scenario, the **zone** might not contain the **fan**.

Try again

Back

Next

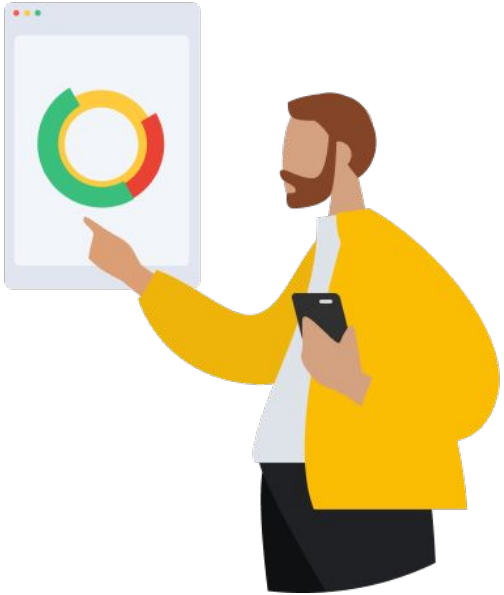
Lesson 8 summary

Let's review what you learned about:

- Connections
- Connection definitions
- Source and target entities
- Connection construction syntax

Now you should be able to:

- Describe the concept of a connection.
- Identify a connection in source code.
- Recognize the different connection definitions.
- Understand the relationship between source and target entities.
- Construct a valid connection.



Back

Note: Remember, the examples in this lesson that were marked with glasses 👓 used the human-readable format of the building config. In your own work, connections should always be encoded in the correct building config format using an actual GUID.

Next

You completed Lesson 8!

Now's a great time to take a quick break before starting Lesson 9.

Ready for Lesson 9?

Let's go!

Back

Helpful resources

For future reference, keep these resources easily accessible for technical and procedural questions.

- digitalbuildings / ontology / yaml / resources / connections.yaml
  Contains all of the available connections.
- GUID Generator and Digital Buildings toolkit
  Used with old format config files to create GUIDs and convert to new format.
- digitalbuildings / ontology
  Contains the documentation and configuration files for the DBO.
- digitalbuildings / ontology / docs / building_config.md
  Describes the configuration format for mapping concrete assets to a model.
- Digital Buildings Project GitHub
  Contains source code, tooling, and documentation for the DBO.