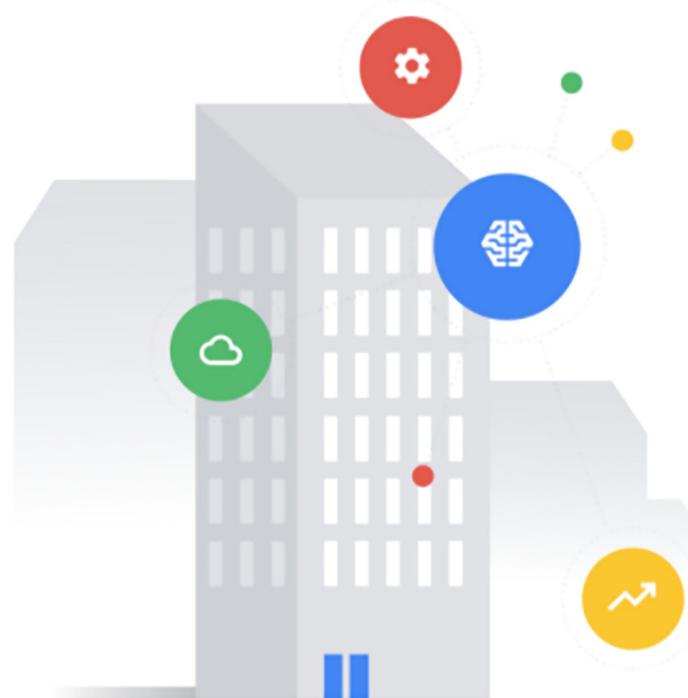




Module 1 | Lesson 7



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!

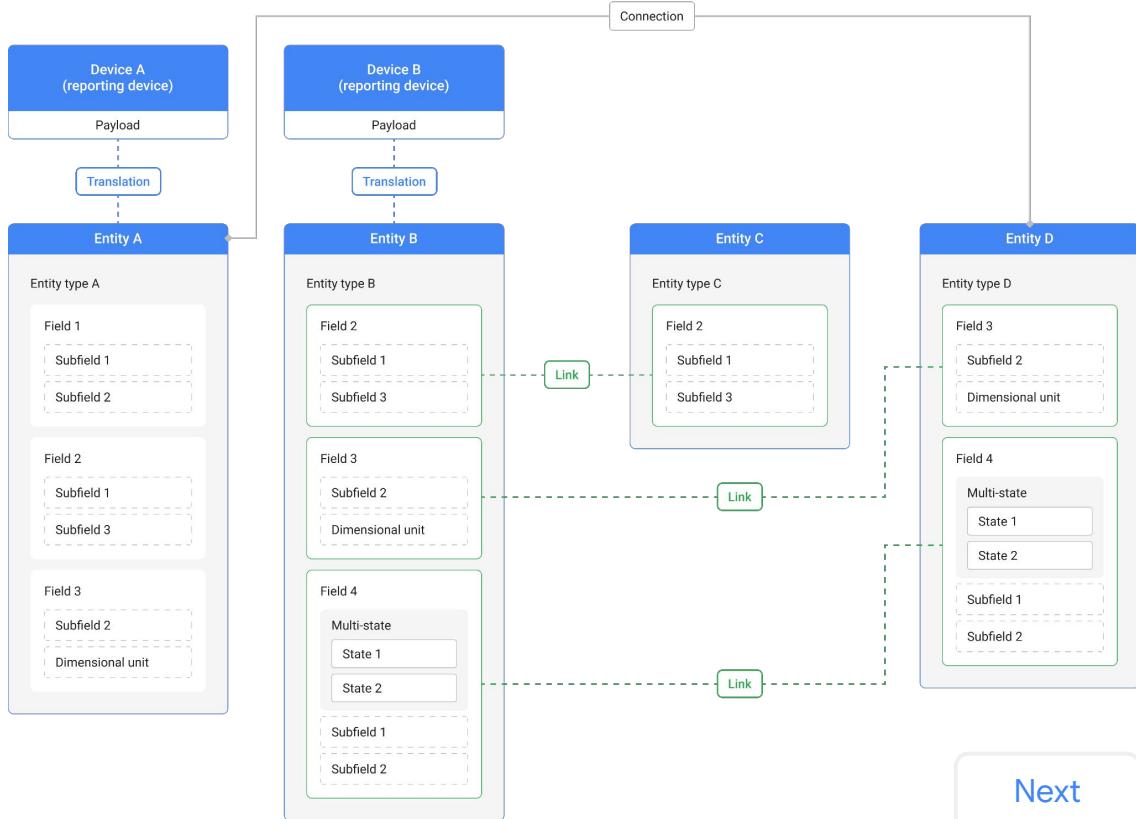
Conceptual model revisited

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

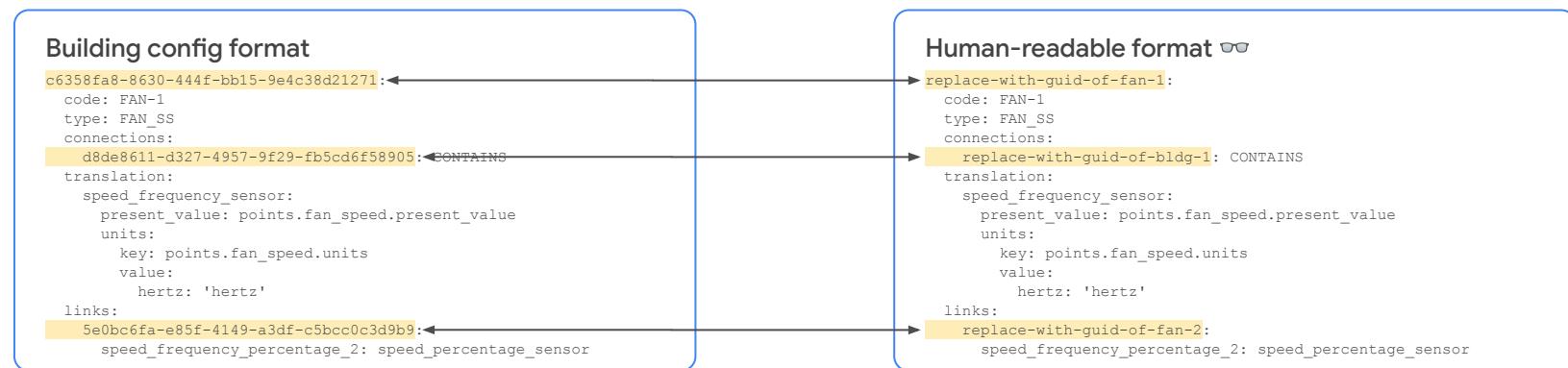
We identify entities using a **globally unique identifier (GUID)** to group their data and link or connect them to other entities.

Mappings are encoded using the **building config format**. The example below shows mappings in the new building config format and highlights the GUIDs of the entity, its connection, and its link. 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.



[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 7.1

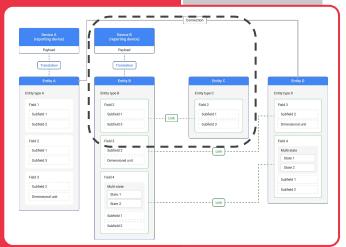
Mappings

What you'll learn about:

- Mappings
- Logical entities

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

- Describe the purpose of mapping.
- Recognize different types of mappings.
- Describe what is a logical entity.



Device B
(reporting device)

Payload

Translation

Entity B

Entity type B

Field 2

Subfield 1

Subfield 3

Field 3

Subfield 2

Dimensional unit

Field 4

Entity C

Entity type C

Field 2

Subfield 1

Subfield 3

Link

Link

Back

Next

What's a mapping?

A mapping shows how a payload should be interpreted by associating the equipment installed in a building with the modeling concepts in the DBO.

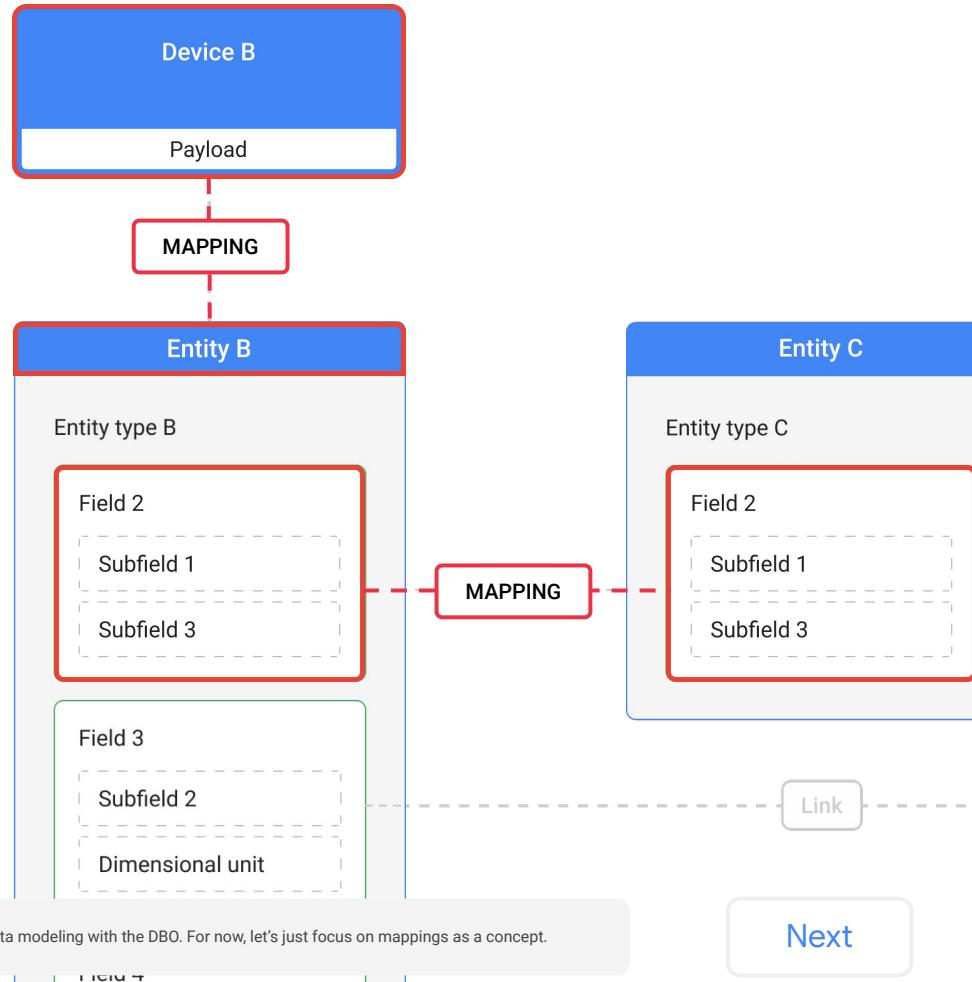
Logical entities

A **logical entity** (also known as a canonical entity) is any device, system, or entity that maps one-to-one with a canonical entity type in the DBO.

You'll always focus on logical entities when mapping devices or systems to the DBO. Simply put, logical entities are the things we care to model, such as fans or meters.

Mappings can be applied in different ways depending on what we care to model.

- Mappings can associate or translate the data from a device's payload to an entity (i.e., a concrete modeling concept) and the fields of its entity type (i.e., an abstract modeling concept).
- Mappings can also be used to pass or link data from one entity and the fields of its entity type to another.



[Back](#)

[Next](#)

Types of mappings

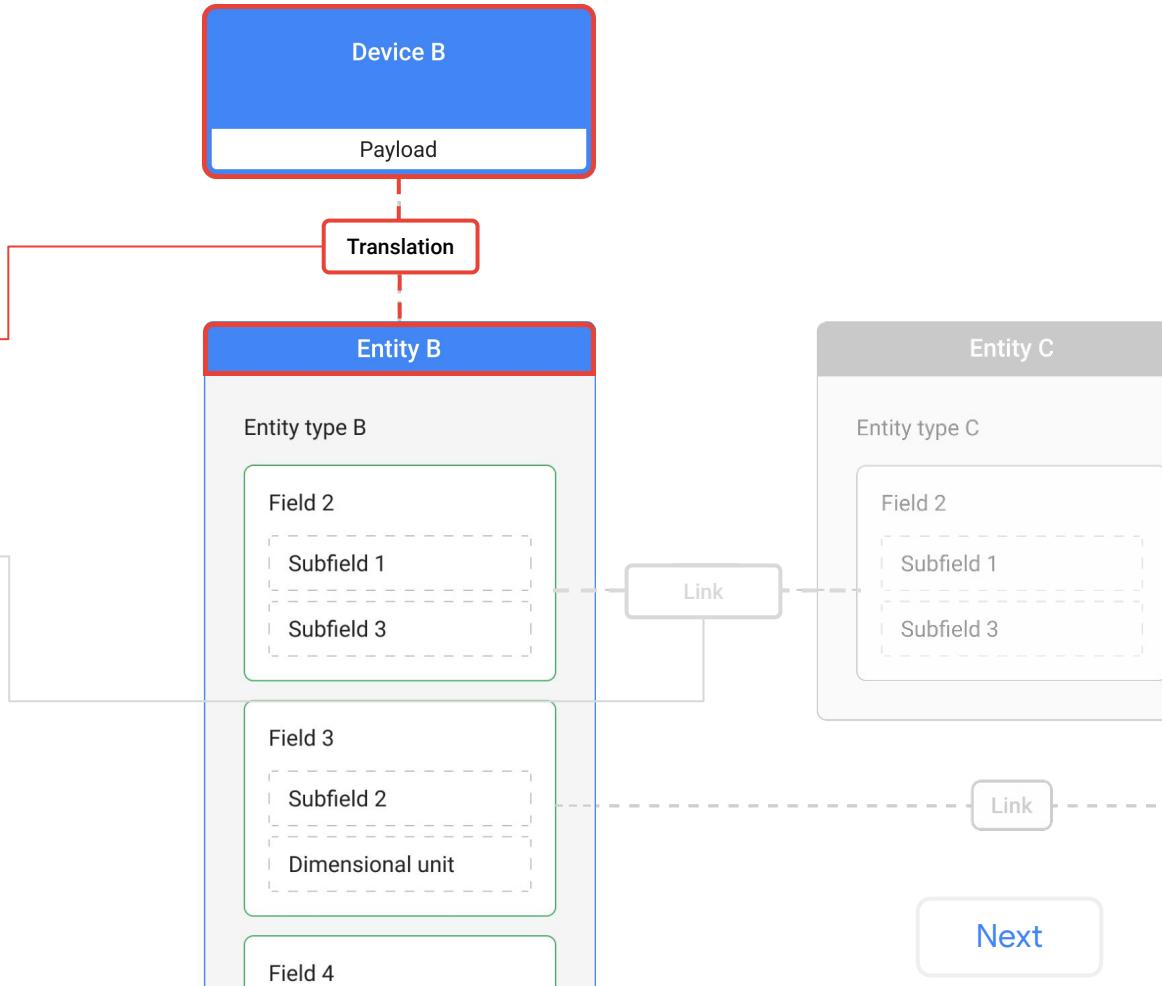
There are two types of mappings in the DBO that are used to associate building equipment to abstract and canonical concepts.

Translations

A translation is a mapping between a device in the real world and its corresponding concrete and abstract concepts in the DBO. Translations convert the information contained in the device's native data payload into the DBO format.

Links

A link is a mapping between the standard fields of two entities to pass data between them. Links are used in conjunction with translations whenever a device's native payload can't be mapped one-to-one with a single abstract concept in the DBO.

[Back](#)[Next](#)

Types of mappings

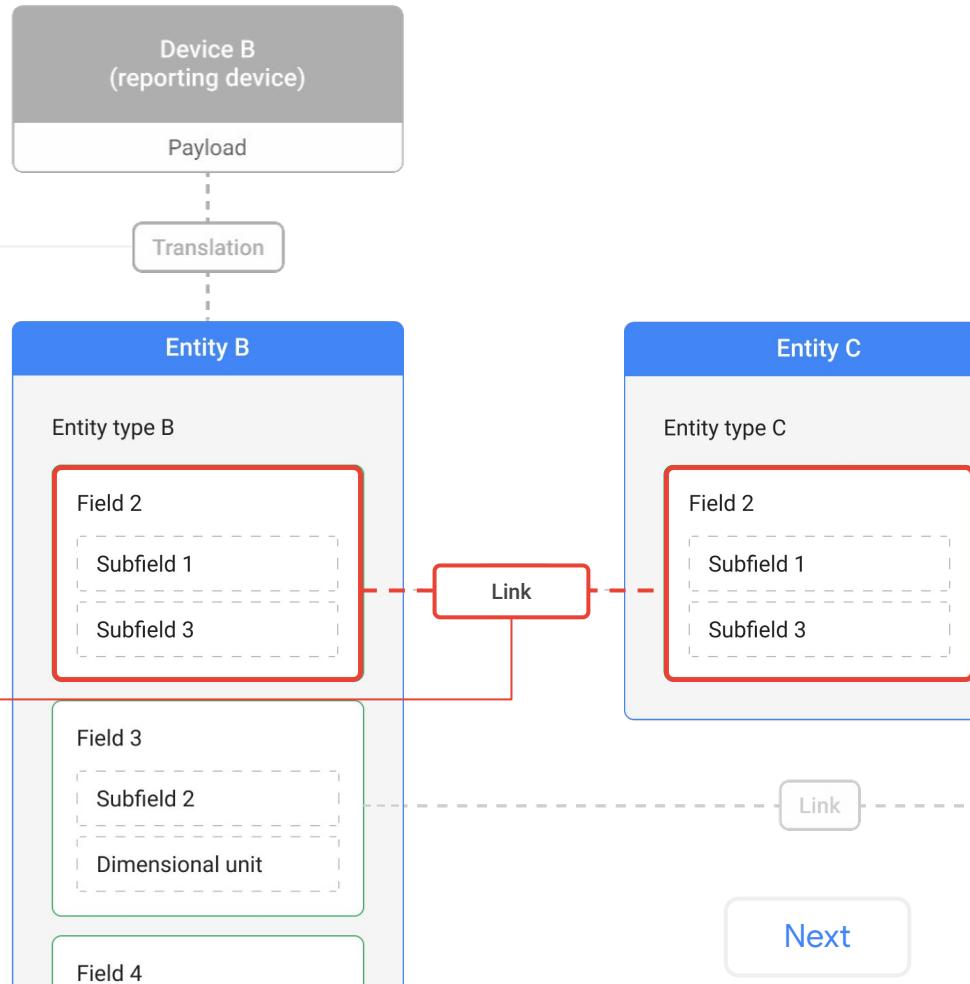
There are two types of mappings in the DBO that are used to associate building equipment to abstract and canonical concepts.

Translations

A translation is a mapping between a device in the real world and its corresponding concrete and abstract concepts in the DBO. Translations convert the information contained in the device's native data payload into the DBO format.

Links

A link is a mapping between the standard fields of two entities to pass data between them. Links are used in conjunction with translations whenever a device's native payload cannot be mapped one-to-one with a single abstract concept in the DBO.

[Back](#)[Next](#)



Lesson 7.2

Translations

What you'll learn about:

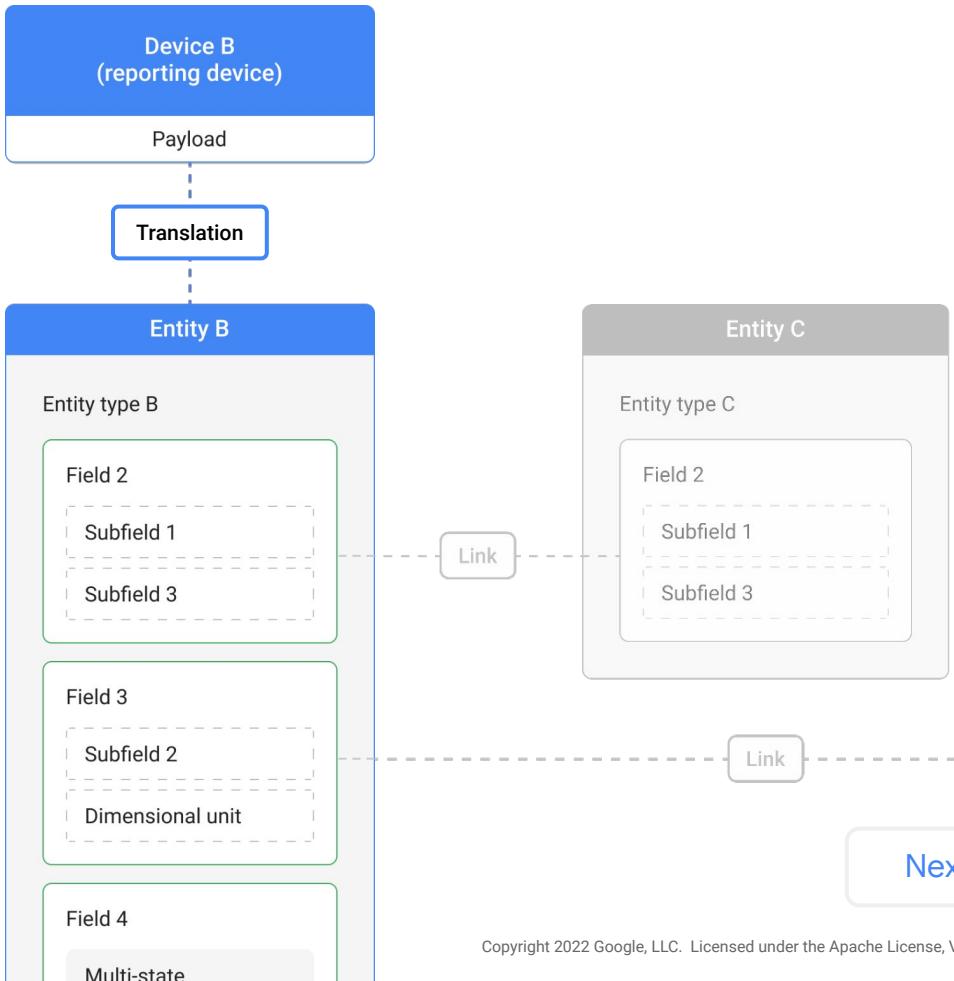
- Translation configuration
- Reporting devices
- Reporting entities
- Point mapping

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

- Recognize the correct configuration of a translation.
- Understand the relationship between reporting devices, reporting entities, and translations.
- Describe the need for point mapping.

[Back](#)

Google



[Next](#)

Translations

A **translation** is a mapping between a device in the real world and its corresponding concepts in the DBO.

Translations convert the information contained in the device's native data payload into the DBO format.

To the right are samples of an actual payload and its translation.

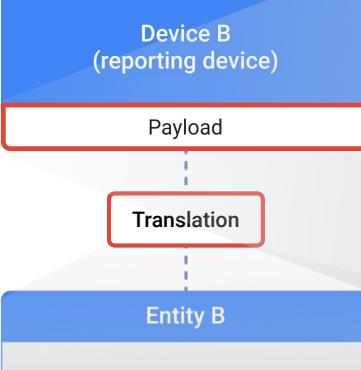
Do you see the point `supply_temp` in the payload sample?

How about in the translation sample?

Note: The translation sample is an abridged version of the full code. The ... indicates abbreviations in the code. Moving forward, you'll notice most samples in this lesson are shortened using

[Back](#)

Google



Payload

```
{  
  "timestamp": "2021-10-18T09:52:43.000Z",  
  "version": 1,  
  "points": [  
    "supply_temp": {  
      "present_value": 23.304852,  
      "units": "degrees-C"  
    },  
    ...  
  ]  
}
```

Translation

```
replace-with-guid-of-fcu-1:  
  code: FCU-1  
  connections:  
    replace-with-guid-of-bldg-1: CONTAINS  
  type: HVAC/FCU_DFSS_...  
  translation:  
    discharge_air_temperature_sensor:  
      present_value: points.supply_temp.present_value  
      units:  
        key: points.supply_temp.units  
        values:  
          degrees_celsius: 'degrees-C'  
    ...
```

Field 4

Multi-state

[Next](#)

Translation configuration

Let's break down this translation.

First, you'll see the entity GUID. This is how we identify entities in the building config format. Remember, we're using simplified GUIDs for these examples!

Immediately below the GUID is the "code" for the entity. This is its human-readable name.

This entity coded **fcu-1** is the reporting entity, which is the concrete modeling concept what represents the reporting device that sends data.

```
Translation
replace-with-guid-of-fcu-1:
  code: FCU-1
  connections:
    replace-with-guid-of-bldg-1: CONTAINS
  type: HVAC/FCU_DFSS_...
  translation:
    discharge_air_temperature_sensor:
      present_value: points.supply_temp.present_value
      units:
        key: points.supply_temp.units
      values:
        degrees_celsius: 'degrees-C'
  ...
  ...
```

Back

Next

Translation configuration (continued)

Let's break down this translation.

Next, you'll notice a few abstract modeling concepts.

There's the entity type, which represents all of the abstract concepts that describe this reporting entity.

There's a field listed inside the `translation` block. This field is from the DBO that maps to specific points from the device's native payload.

```
Translation ▾
replace-with-guid-of-fcu-1:
  code: FCU-1
  connections:
    replace-with-guid-of-bldg-1: CONTAINS
      ➔ type: HVAC/FCU_DFSS_...
      ➔ discharge_air_temperature_sensor:
        present_value: points.supply_temp.present_value
        units:
          key: points.supply_temp.units
        values:
          degrees_celsius: 'degrees-C'
  ...
  ...
```

[Back](#)

[Next](#)

Translation configuration (continued)

Let's break down this translation.

Don't mistake some things for a field inside the field block!

Here's the DBO standard field.

Highlighted inside the field block are specific points from the reporting device's native payload. These points are what's converted into the DBO format using a translation.

That means in this sample, the point `supply_temp` is translated to the field `discharge_air_temperature_sensor`.

Translation ☰

```
replace-with-guid-of-fcu-1:  
  code: FCU-1  
  connections:  
    replace-with-guid-of-bldg-1: CONT  
  type: HVAC/FCU_DFSS_...  
  translation:  
    discharge_air_temperature_sensor:  
      present_value: points supply_temp.present_value  
      units:  
        key: points.supply_temp.units  
        values:  
          degrees_celsius: 'degrees-C'  
    ...
```

Payload

```
{  
  "timestamp": "2021-10-18T09:52:43.000Z",  
  "version": 1,  
  "points": {  
    "supply_temp": {  
      "present_value": 23.304852,  
      "units": "degrees-C"  
    },  
    ...  
  }  
}
```

Back

Next

Translation configuration (continued)

Let's break down this translation.

Also inside the field block is information about the field's specific dimensional units or states. In this sample, only units are specified because it isn't multi-state.

Finally, notice the . separator used to indicate the access path to the nested keys in the JSON payload.

Translation

```
replace-with-guid-of-fcu-1:  
  code: FCU-1  
  connections:  
    replace-with-guid-of-bldg-1: CONT  
  type: HVAC/FCU_DFSS_...  
  translation:  
    discharge_air_temperature_sensor:  
      present_value: points.supply_temp.present_value  
      units:  
        key: points.supply_temp.units  
        values:  
          degrees_celsius: 'degrees-C'  
    ...
```

Payload

```
{  
  "timestamp": "2021-10-18T09:52:43.000Z",  
  "version": 1,  
  "points": {  
    "supply_temp": {  
      "present_value": 23.304852,  
      "units": "degrees-C"  
    },  
    ...  
  }  
}
```

Back

Next

Mapping with translations

A **translation** maps the native payload of a **reporting device** to the standard fields of a **reporting entity**.

What's a reporting device?

A **reporting device** is any device or system that generates and sends a payload of data to the cloud.

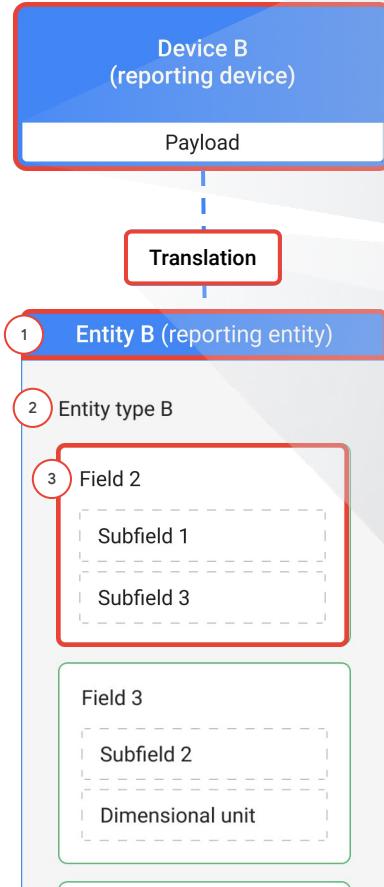
Some examples include:

- A controller for an individual device
- A network controller for multiple devices

What's a reporting entity?

A **reporting entity** is the concrete instance of the reporting device expressed in the building configuration file.

Like all entities, a reporting entity has an entity type, which groups fields and other abstract modeling concepts that describe its properties.



Payload

```
{  
  "timestamp": "2021-10-18T09:52:43.000Z",  
  "version": 1,  
  "points": {  
    "supply_temp": {  
      "present_value": 23.304852,  
      "units": "degrees-C"  
    },  
    ...  
  }  
}
```

Translation

```
1 replace-with-guid-of-fcu-1:  
  code: FCU-1  
  connections:  
    replace-with-guid-of-bldg-1: CONTAINS  
  2 type: HVAC/FCU_DFSS_...  
  translation:  
    3 discharge_air_temperature_sensor:  
      present_value:  
      points.supply_temp.present_value  
      units:  
        key: pointset.points.supply_temp.units  
        values:  
          degrees_celsius: 'degrees-C'  
  ...
```

Back

Next

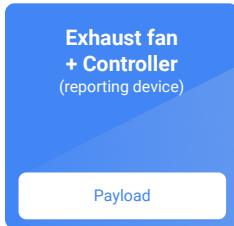
Mapping with translations (continued)

A **translation** maps the native payload of a **reporting device** to the standard fields of a **reporting entity**.

Example

Let's say you have an exhaust fan with a controller that you want to model.

The controller sends the fan's native payload to the cloud, which makes it a reporting device. However, the cloud does not recognize its native payload.



Payload

```
{  
  "timestamp": "2021-10-18T09:52:43.000Z",  
  "points": [  
    {"fan_speed": {  
      "present_value": 60,  
      "units": "hertz"  
    }},  
    {"fan_pwr": {  
      "present_value": 12.77,  
      "units": "kw"  
    }}  
  ]  
}
```

[Back](#)

Click **Next** to inspect the translation in building config format (human-readable).

[Next](#)

Mapping with translations (continued)

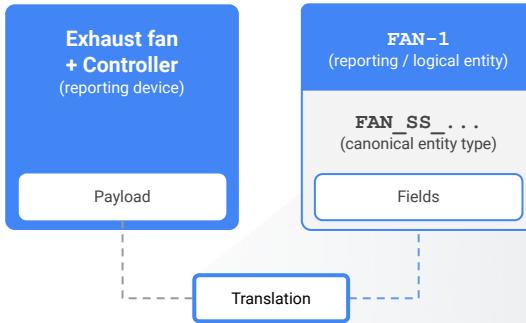
A **translation** maps the native payload of a **reporting device** to the standard fields of a **reporting entity**.

Example (continued)

The exhaust fan is a type of **FAN_SS_...**, which is a canonical entity type. We'll generate a GUID to properly identify the reporting entity and reference it with the code **FAN-1**.

Using a translation, you can map the payload to the reporting entity **FAN-1** and the standard fields of **FAN_SS_...**. Since this is a one-to-one mapping, **FAN-1** is also a logical entity.

Now the exhaust fan is able to send its translated payload, and the cloud can recognize it.



Translation 🔍

```
replace-with-guid-of-fan-1:  
  code: FAN-1  
  type: FAN_SS_...  
  translation:  
    speed_frequency_sensor:  
      present_value: points.fan_speed.present_value  
      units:  
        key: points.fan_speed.units  
        value:  
          hertz: 'hertz'  
    power_sensor:  
      present_value: points.fan_pwr.present_value  
      units:  
        key: points.fan_pwr.units  
        value:  
          kilowatts: 'kw'
```

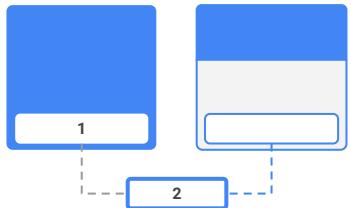
[Back](#)

Click **Next** for a summary of this example.

[Next](#)

Mapping with translations (continued)

A **translation** maps the native payload of a **reporting device** to the standard fields of a **reporting entity**.



Example summary

This mapping requires the following entities:

- One logical entity
- One reporting entity
- One canonical entity

In this particular example, the logical, reporting, and canonical entities are all **FAN-1**.

1. Payload

```
{  
  "timestamp": "2021-10-18T09:52:43.000Z",  
  "points": {  
    "fan_speed": {  
      "present_value": 60,  
      "units": "hertz"  
    },  
    "fan_pwr": {  
      "present_value": 12.77,  
      "units": "kw"  
    }  
  }  
}
```

2. Translation ☺

```
replace-with-guid-of-fan-1:  
  code: FAN-1  
  type: FAN_SS_...  
  translation:  
    speed_frequency_sensor:  
      present_value: points. fan_speed.present_value  
      units:  
        key: points. fan_speed.units  
        value:  
          hertz: 'hertz'  
    power_sensor:  
      present_value: points. fan_pwr.present_value  
      units:  
        key: points. fan_pwr.units  
        value:  
          kilowatts: 'kw'
```

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, mappings should always be encoded in the correct building config format using an actual GUID.

Next

Point mapping with translations

A translation “tells” the system exactly which points from the native payload that we care about.

A reporting device can send a lot of points in its payload. However, not every point needs to be mapped.

Using translations, you can pinpoint useful data points from a reporting device’s payload to map to the fields of a reporting entity. This allows you to omit unnecessary points.

Example

Let’s say you have a controller that collects data for a fan.

The controller sends all of this data to the cloud in a single payload, which makes it a reporting device. However, you don’t care to map the `status_alarm` data point.

The fan is a type of `FAN_SS...`, which is a canonical entity type. We’ll generate a GUID to identify the reporting entity and name it with the code `FAN_3`.

Using a translation, you can choose to map only the useful points from the payload and omit unnecessary ones like `status_alarm`.

Now the fan is able to send its translated payload, and the cloud can recognize it.

Payload

```
{  
  "timestamp": "2021-10-18T09:52:43.000Z",  
  "points": {  
    "fan_speed": {  
      "present_value": 60,  
      "units": "hertz"  
    },  
    "status_alarm": {  
      "present_value": OFF  
    }  
  }  
}
```

Translation

```
replace-with-guid-of-fan-3:  
  code: FAN_3  
  type: FAN_SS_...  
  translation:  
    speed_frequency_sensor:  
      present_value: pointsfan_speed.present_value  
      units:  
        key: pointsfan_speed.units  
        value:  
          hertz: 'hertz'  
  #NOTE status_alarm isn't translated, because not  
  #everything in the payload must be translated.
```

Back

Note: You’ll learn more about required device data and determining what needs to be modeled in Module 2: Data modeling with the DBO.

Next

Lesson 7

Knowledge check 1



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

- The next slide will have a question about translations.
- Review the question and select the correct response.
- After this knowledge check, you'll move on to learn about mapping with links.

You won't be able to move forward until the correct answer is selected.

[Back](#)

[Next](#)

Knowledge check 1

A sample payload is on the right.

Given this payload,
which translation is configured correctly?

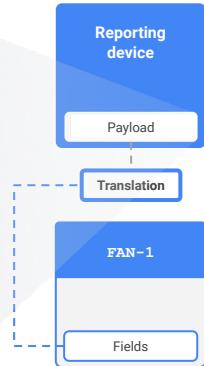
Select the best answer from the options listed below.

```
replace-with-guid-of-fan-1:  
  code: FAN-1  
  translation:  
    speed_frequency_sensor:  
      present_value: points.fan_speed.present_value  
      units:  
        key: points.fan_speed.units  
        value:  
          hertz: 'hertz'  
    power_sensor:  
      present_value: points.fan_pwr.present_value  
      units:  
        key: points.fan_pwr.units  
        value:  
          kilowatts: 'kw'
```

```
replace-with-guid-of-fan-1:  
  code: FAN-1  
  type: FAN_SS_...  
  translation:  
    speed_frequency_sensor:  
      present_value: points.fan_speed.present_value  
      units:  
        key: points.fan_speed.units  
        value:  
          hertz: 'hertz'  
    power_sensor:  
      present_value: points.fan_pwr.present_value  
      units:  
        key: points.fan_pwr.units  
        value:  
          kilowatts: 'kw'
```

```
replace-with-guid-of-fan-1:  
  code: FAN-1  
  type: FAN_SS_...  
  translation:  
    speed_frequency_sensor:  
      present_value: points.fan_pwr.present_value  
      units:  
        key: points.fan_pwr.units  
        value:  
          hertz: 'hertz'  
    power_sensor:  
      present_value: points.fan_speed.present_value  
      units:  
        key: points.fan_speed.units  
        value:  
          kilowatts: 'kw'
```

```
{  
  "timestamp": "2021-10-18T09:52:43.000Z",  
  "points": {  
    "fan_speed": {  
      "present_value": 60,  
      "units": "hertz"  
    },  
    "fan_pwr": {  
      "present_value": 12.77,  
      "units": "kw"  
    }  
  }  
}
```



Back

Note: Don't worry! We've already provided the standard fields for you. For this question, pay attention to the configuration of each option based on the information you're given from the sample. Also note that the translation options are shown in the human-readable format (☞) of the building config.

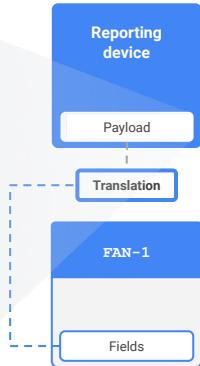
Next

Hmm, that's not right! 🤔

Look closely! This translation is missing an entity type.
Remember, an entity should always have an entity type.

Try again

```
{  
  "timestamp": "2021-10-18T09:52:43.000Z",  
  "points": {  
    "fan_speed": {  
      "present_value": 60,  
      "units": "hertz"  
    },  
    "fan_pwr": {  
      "present_value": 12.77,  
      "units": "kw"  
    }  
  }  
}
```



```
replace-with-guid-of-fan-1:  
  code: FAN-1  
  translation:  
    speed_frequency_sensor:  
      present_value: points.fan_speed.present_value  
      units:  
        key: points.fan_speed.units  
        value:  
          hertz: 'hertz'  
    power_sensor:  
      present_value: points.fan_pwr.present_value  
      units:  
        key: points.fan_pwr.units  
        value:  
          kilowatts: 'kw'
```

```
replace-with-guid-of-fan-1:  
  code: FAN-1  
  type: FAN_SS_...  
  translation:  
    speed_frequency_sensor:  
      present_value: points.fan_speed.present_value  
      units:  
        key: points.fan_speed.units  
        value:  
          hertz: 'hertz'  
    power_sensor:  
      present_value: points.fan_pwr.present_value  
      units:  
        key: points.fan_pwr.units  
        value:  
          kilowatts: 'kw'
```

```
replace-with-guid-of-fan-1:  
  code: FAN-1  
  type: FAN_SS_...  
  translation:  
    speed_frequency_sensor:  
      present_value: points.fan_pwr.present_value  
      units:  
        key: points.fan_pwr.units  
        value:  
          hertz: 'hertz'  
    power_sensor:  
      present_value: points.fan_speed.present_value  
      units:  
        key: points.fan_speed.units  
        value:  
          kilowatts: 'kw'
```

Back

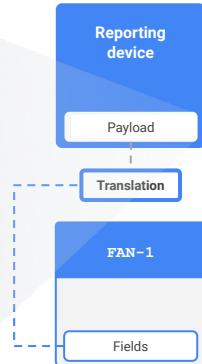
Next

That's right! 🎉

This translation does the following:

- The point `fan_speed` translates to the field `speed_frequency_sensor`.
- The point `fan_pwr` translates to the field `power_sensor`.

```
{  
    "timestamp": "2021-10-18T09:52:43.000Z",  
    "points": {  
        "fan_speed": {  
            "present_value": 60,  
            "units": "hertz"  
        },  
        "fan_pwr": {  
            "present_value": 12.77,  
            "units": "kw"  
        }  
    }  
}
```



```
replace-with-guid-of-fan-1:  
  code: FAN-1  
  translation:  
    speed_frequency_sensor:  
      present_value: points.fan_speed.present_value  
      units:  
        key: points.fan_speed.units  
        value:  
          hertz: 'hertz'  
    power_sensor:  
      present_value: points.fan_pwr.present_value  
      units:  
        key: points.fan_pwr.units  
        value:  
          kilowatts: 'kw'
```

```
replace-with-guid-of-fan-1:  
  code: FAN-1  
  type: FAN_SS_...  
  translation:  
    speed_frequency_sensor:  
      present_value: points.fan_speed.present_value  
      units:  
        key: points.fan_speed.units  
        value:  
          hertz: 'hertz'  
    power_sensor:  
      present_value: points.fan_pwr.present_value  
      units:  
        key: points.fan_pwr.units  
        value:  
          kilowatts: 'kw'
```

```
replace-with-guid-of-fan-1:  
  code: FAN-1  
  type: FAN_SS_...  
  translation:  
    speed_frequency_sensor:  
      present_value: points.fan_pwr.present_value  
      units:  
        key: points.fan_pwr.units  
        value:  
          hertz: 'hertz'  
    power_sensor:  
      present_value: points.fan_speed.present_value  
      units:  
        key: points.fan_speed.units  
        value:  
          kilowatts: 'kw'
```

[Back](#)

Click [Next](#) to move on to and learn about mapping with links.

[Next](#)

Hmm, that's not right! 🤔

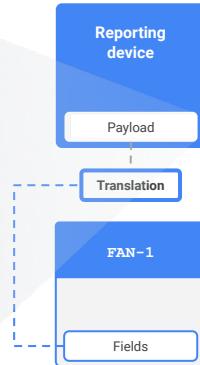
Look closely! It doesn't make much sense to translate the point `fan_pwr` to the field `speed_frequency_sensor`.

Try again

```
replace-with-guid-of-fan-1:  
  code: FAN-1  
  translation:  
    speed_frequency_sensor:  
      present_value: points.fan_speed.present_value  
      units:  
        key: points.fan_speed.units  
        value:  
          hertz: 'hertz'  
    power_sensor:  
      present_value: points.fan_pwr.present_value  
      units:  
        key: points.fan_pwr.units  
        value:  
          kilowatts: 'kw'
```

```
replace-with-guid-of-fan-1:  
  code: FAN-1  
  type: FAN_SS_...  
  translation:  
    speed_frequency_sensor:  
      present_value: points.fan_speed.present_value  
      units:  
        key: points.fan_speed.units  
        value:  
          hertz: 'hertz'  
    power_sensor:  
      present_value: points.fan_pwr.present_value  
      units:  
        key: points.fan_pwr.units  
        value:  
          kilowatts: 'kw'
```

```
{  
  "timestamp": "2021-10-18T09:52:43.000Z",  
  "points": {  
    "fan_speed": {  
      "present_value": 60,  
      "units": "hertz"  
    },  
    "fan_pwr": {  
      "present_value": 12.77,  
      "units": "kw"  
    }  
  }  
}
```



```
replace-with-guid-of-fan-1:  
  code: FAN-1  
  type: FAN_SS_...  
  translation:  
    speed_frequency_sensor:  
      present_value: points.fan_pwr.present_value  
      units:  
        key: points.fan_pwr.units  
        value:  
          hertz: 'hertz'  
    power_sensor:  
      present_value: points.fan_speed.present_value  
      units:  
        key: points.fan_speed.units  
        value:  
          kilowatts: 'kw'
```

Back

Next



Lesson 7.3

Links

What you'll learn about:

- Link configuration
- Virtual entities
- Passthrough entities

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

- Recognize the correct configuration of a link.
- Determine when a virtual entity is needed.
- Understand the dependencies between links and translations for data modeling.
- Determine when a passthrough entity is needed.

[Back](#)

Google



[Next](#)

Links

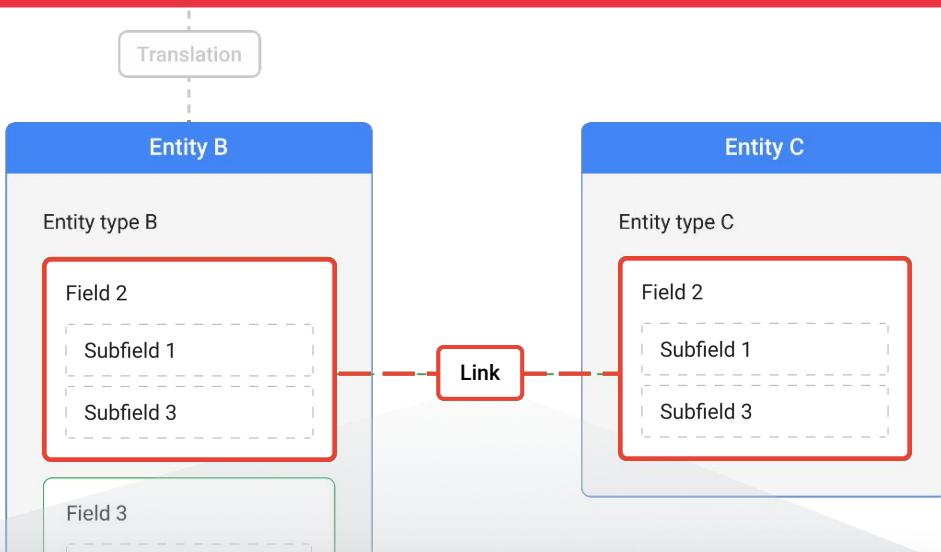
A **link** is a mapping between the standard fields of two entity types.

Links are used together with translations whenever a device's native payload can't be mapped one-to-one with a single abstract concept in the DBO.

To the right is a sample of a link. You'll notice there isn't a sample payload, though.

Can you think of why there isn't a sample payload here?

If not, that's alright! You'll find out in a few slides.



Link 🔗

```
replace-with-guid-of-vav-32: #NOTE This is the target device.
```

```
code: VAV-32
```

```
type: HVAC/VAV_SD_DSP_CO2C
```

links:

```
replace-with-guid-of-gtwy-1:: #NOTE This is the source device.
```

```
supply_air_damper_percentage_command:supply_air_damper_percentage_command_1  
#NOTE The order of fields = target_device_field:source_device_field
```

State 2

```
Subfield 1
```

```
Subfield 2
```

Back

Next

Link configuration

Let's break down this link.

First, you'll see the entity GUID followed by the entity's "code." This is the target entity, which defines the link.

Inside the `links` block, you'll see the GUID of another entity. This is the source entity, which lists its fields that are linked to the target entity.

```
Link
replace-with-guid-of-vav-32 #NOTE This is the target device.
  code: VAV-32
  type: HVAC/VAV_SD_DSP_CO2C
  links:
    replace-with-guid-of-gtwy-1:#NOTE This is the source device.
      supply_air_damper_percentage_command:supply_air_damper_percentage_command_1
      #NOTE The order of fields = target_device_field:source_device_field
    ...
```

Back

Note: The concept of source and target entities is also important for connections, which you'll learn about in [Lesson 8: Connections](#).

Next

Link configuration (continued)

Let's break down this link.

Next, you'll notice a few abstract modeling concepts.

There's the entity type, which represents all of the abstract concepts that describe the target entity.

In the source entity block, you'll see lines with two fields.

Link

```
replace-with-guid-of-vav-32: #NOTE This is the target device.
```

```
  code: VAV-32
```

```
  ➔ type: HVAC/VAV_SD_DSP_CO2C
```

```
  links:
```

```
    replace-with-guid-of-gtwy-1: #NOTE This is the source device.
```

```
      ➔ supply_air_damper_percentage_command:supply_air_damper_percentage_command_1
```

```
      #NOTE The order of fields = target_device_field:source_device_field
```

```
  ...
```

The first field is from the target entity.

The second field is from the source entity.

[Back](#)

[Next](#)

Mapping with links

Linking the standard fields of one entity to the standard fields of another entity results in a **virtual entity**.

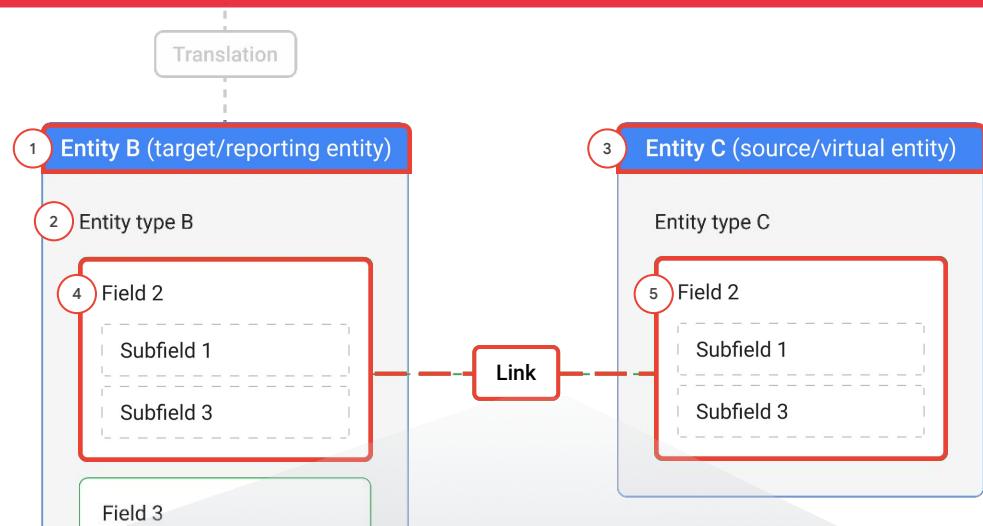
What's a virtual entity?

A **virtual entity** is a representation of a logical entity constructed by linking the fields of a reporting entity to the fields of a logical entity.

Why are virtual entities needed?

Virtual entities are needed any time a reporting device can't map one-to-one with a single logical entity. Some use cases include:

- You want to include data from one reporting device in multiple logical entities.
- You want to model a reporting device with multiple components that send their payload via the same network controller.



Link 🔗

```
1 replace-with-guid-of-vav-32: #NOTE This is the target device.  
  code: VAV-32  
2 type: HVAC/VAV_SD_DSP_CO2C  
  links:  
    3 replace-with-guid-of-gtwy-1: #NOTE This is the source device.  
    4 supply_air_damper_percentage_command:supply_air_damper_percentage_command_1  
      #NOTE The order of fields = target_device_field : source_device_field
```

Back

Google

Next

Mapping with links and translations

Links are used in conjunction with **translations** to map the payload of a **reporting device** to the fields of a **virtual entity**.

So, why isn't there a payload?

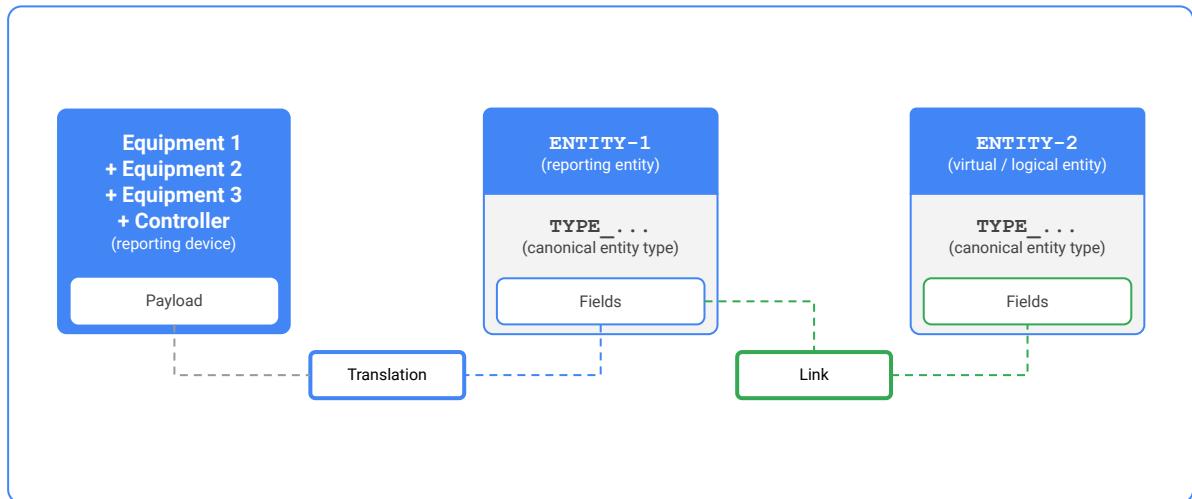
Compared to a translation, links don't deal with the reporting device's payload directly.

Links only map fields to other fields. And remember, fields by themselves don't deal with the payload. They need to be mapped to a device's payload and its specific points with a translation.

Links can't do the job alone

Remember, the purpose of mapping is to associate the native payload of a building's equipment with the abstract and canonical concepts of the DBO.

Since links only map fields to other fields, there must be a valid translation between a reporting device's payload and a reporting entity's fields. This can result in many different ways to combine links and translations.



[Back](#)

Note: A model using only a translation mapping can be valid. However, a model only using a link mapping is never valid.

[Next](#)

Mapping with links and translations (continued)

Links are used in conjunction with **translations** to map the payload of a **reporting device** to the fields of a **virtual entity**.

Example

Let's say you have a fan coil unit (FCU) with a zone temperature (ZNT) sensor and a single controller. You want to model the ZNT sensor as its own device because it's in a different room than the FCU.

The FCU's controller sends the device's native payload to the cloud, which makes it a reporting device. However, the cloud doesn't recognize its native payload.



Payload

```
{  
  "timestamp": "2021-10-18T09:52:43.000Z",  
  "points": [  
    "zone_temperature": {  
      "present_value": 72.43,  
      "units": "degrees-F"  
    },  
    "zone_temperature_setpoint": {  
      "present_value": 71,  
      "units": "degrees-F"  
    },  
    "discharge_temp": {  
      "present_value": 55.232,  
      "units": "degrees-F"  
    },  
    "status_alarm": {  
      "present_value": OFF  
    }  
  ]  
}
```

[Back](#)

Click [Next](#) to inspect the translation in building config format (human-readable).

[Next](#)

Mapping with links and translations (continued)

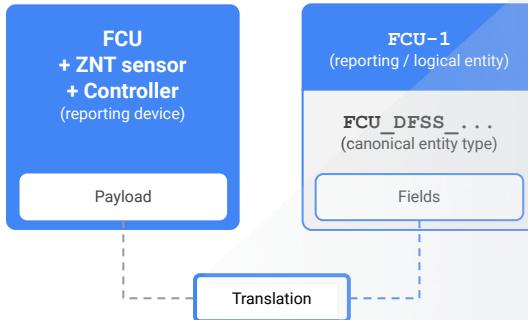
Links are used in conjunction with **translations** to map the payload of a **reporting device** to the fields of a **virtual entity**.

Example (continued)

The FCU is a type of `FCU_DFSS_...`, which is a canonical entity type. We'll generate a GUID to properly identify the reporting entity and reference it with the code `FCU-1`.

Using a translation, you can map the payload to the reporting entity `FCU-1` and standard fields of `FCU_DFSS_...`. Since this is a one-to-one mapping, `FCU-1` also a logical entity.

Now the FCU can send its translated payload, and the cloud can recognize it. However, the ZNT isn't modeled as its own device. It's just a field living on the FCU.



Translation ☰

```
replace-with-guid-of-fcu-1:  
  code: FCU-1  
  type: FCU_DFSS_... #NOTE This is a logical entity with a curated type.  
  translation:  
    zone_air_temperature_sensor:  
      present_value: points.zone_temperature.present_value  
      units:  
        key: points.zone_temperature.units  
        value:  
          degrees_fahrenheit: 'degrees-F'  
    zone_air_temperature_setpoint:  
      present_value: points.zone_temperature_setpoint.present_value  
      units:  
        key: points.zone_temperature_setpoint.units  
        value:  
          degrees_fahrenheit: 'degrees-F'  
    discharge_air_temperature_sensor:  
      present_value: points.discharge_temp.present_value  
      units:  
        key: points.discharge_temp.units  
        value:  
          degrees_fahrenheit: 'degrees-F'  
  #NOTE status_alarm isn't translated, because not everything in the  
  payload must be translated.  
  ...
```

[Back](#)

Click [Next](#) to inspect the link in building config format (human-readable).

[Next](#)

Mapping with links and translations (continued)

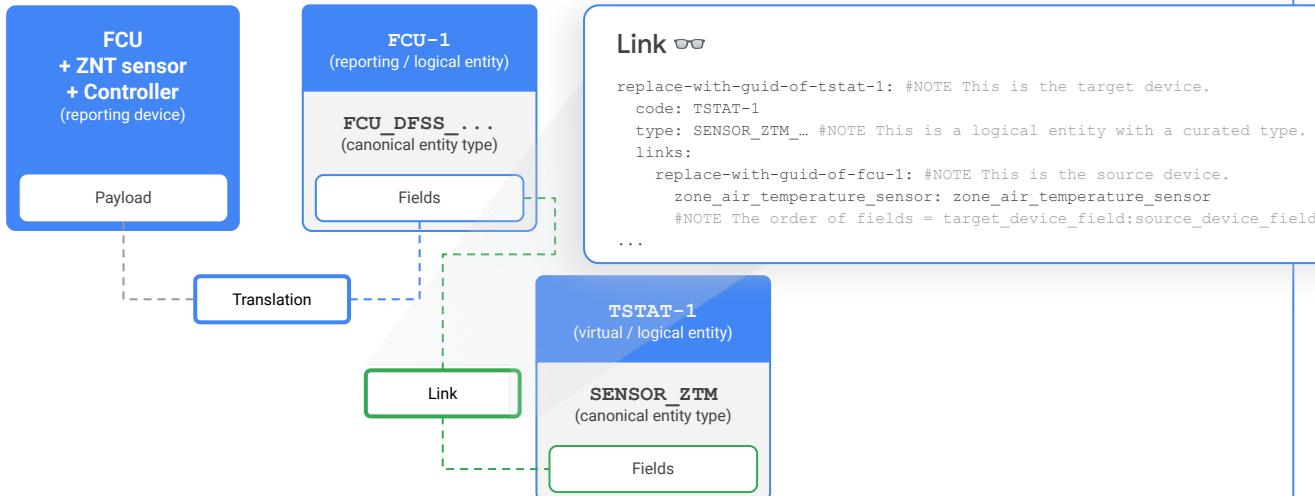
Links are used in conjunction with **translations** to map the payload of a **reporting device** to the fields of a **virtual entity**.

Example (continued)

The ZNT sensor is a type of `SENSOR_ZTM` and a canonical entity type. Since it's something we care to model, we'll generate a GUID to properly identify the virtual entity and reference it with the code `TSTAT-1`.

However, the ZNT sensor can't map one-to-one to the fields of `SENSOR_ZTM`, because the reporting device's payload is already mapped to `FCU-1`.

Using links, you can map the fields of the reporting entity `FCU-1` to the fields of the virtual entity `TSTAT-1`. Now these entities share the same ZNT sensor value.

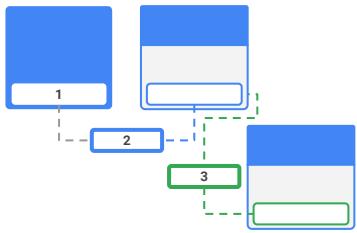


[Back](#)

Click [Next](#) for a summary of this example.

[Next](#)

Mapping with links and translations (continued)



Example summary

This mapping requires the following:

- Two logical entities: **FCU-1**, **TSTAT-1**
 - One reporting entity: **FCU-1**
 - One virtual entity: **TSTAT-1**
- Two canonical entity types: **FCU_DFSS_...**, **SENSOR_ZTM_...**

1. Payload

```
{  
  "timestamp": "2021-10-18T09:52:43.000Z",  
  "points": {  
    "zone_temperature": {  
      "present_value": 72.43,  
      "units": "degrees-F"  
    },  
    "zone_temperature_setpoint": {  
      "present_value": 71,  
      "units": "degrees-F"  
    },  
    "discharge_temp": {  
      "present_value": 55.232,  
      "units": "degrees-F"  
    },  
    "status_alarm": {  
      "present_value": OFF  
    }  
  }  
}
```

2. Translation ☺

```
replace-with-guid-of-fcu-1:  
  code: FCU-1  
  type: FCU_DFSS_... #NOTE This is a logical entity with a curated type.  
  translation:  
    zone_air_temperature_sensor:  
      present_value: points.zone_temperature.present_value  
      units:  
        key: points.zone_temperature.units  
        value:  
          degrees_fahrenheit: 'degrees-F'  
    zone_air_temperature_setpoint:  
      present_value: points.zone_temperature_setpoint.present_value  
      units:  
        key: points.zone_temperature_setpoint.units  
        value:  
          degrees_fahrenheit: 'degrees-F'  
    discharge_air_temperature_sensor:  
      present_value: points.discharge_temp.present_value  
      units:  
        key: points.discharge_temp.units  
        value:  
          degrees_fahrenheit: 'degrees-F'  
#NOTE status_alarm isn't translated, because not everything in the  
payload must be translated.  
...
```

3. Link ☺

```
replace-with-guid-of-tstat-1: #NOTE This is the target device.  
  code: TSTAT-1  
  type: SENSOR_ZTM_... #NOTE This is a logical entity with links:  
  replace-with-guid-of-fcu-1: #NOTE This is the source device.  
    zone_air_temperature_sensor: zone_air_temperature_sensor  
    #NOTE The order of fields = target_device_field:source_device_field  
...
```

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, mappings should always be encoded in the correct building config format using an actual GUID.

Next

Lesson 7

Knowledge check 2



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

- The next slide will have a question about translations.
- Review the question and select the correct response.
- After this knowledge check, you'll move on to learn about mapping with passthrough entities.

You won't be able to move forward until the correct answer is selected.

[Back](#)

[Next](#)

Knowledge check 2

A sample translation is on the right.

Given this translation,
which link is configured correctly?

Select the best answer from the options listed below.

```
replace-with-guid-of-ef-1: #NOTE This is target device.  
  code: EF-1  
  type: FAN_SS_...  
  links:  
    speed_frequency_sensor: speed_frequency_sensor_1  
    power_sensor: power_sensor_1  
...
```

```
replace-with-guid-of-ef-1: #NOTE This is target device.  
  code: EF-1  
  type: FAN_SS_...  
  links:  
    the-gtwy-guid:  
      speed_frequency_sensor_1: speed_frequency_sensor  
      power_sensor_1: power_sensor  
...
```

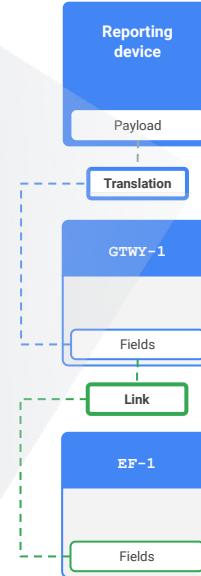
```
replace-with-guid-of-ef-1: #NOTE This is target device.  
  code: EF-1  
  type: FAN_SS_...  
  links:  
    the-gtwy-guid:  
      speed_frequency_sensor: speed_frequency_sensor_1  
      power_sensor: power_sensor_1  
...
```

Back

Note: Don't worry! We've already provided the standard fields for you. For this question, pay attention to the configuration of each option based on the information you're given from the sample. Also note that the translation options are shown in the human-readable format («) of the building config.

Next

```
replace-with-guid-of-gtwy-1:  
  code: GTWY-1  
  type: GATEWAYS/PASSTHROUGH  
  translation:  
    speed_frequency_sensor_1:  
      present_value: points.ef_1_speed.present_value  
      units:  
        key: points.ef_1_speed.units  
        value:  
          hertz: 'hertz'  
    speed_frequency_sensor_2:  
      present_value: points.ef_2_speed.present_value  
      units:  
        key: points.ef_2_speed.units  
        value:  
          hertz: 'hertz'  
    power_sensor_1:  
      present_value: points.ef_1_kw.present_value  
      units:  
        key: points.ef_1_kw.units  
        value:  
          kilowatts: 'kw'  
    power_sensor_2:  
      present_value: points.ef_2_kw.present_value  
      units:  
        key: points.ef_2_kw.units  
        value:  
          kilowatts: 'kw'  
...
```



Hmm, that's not right! 🤔

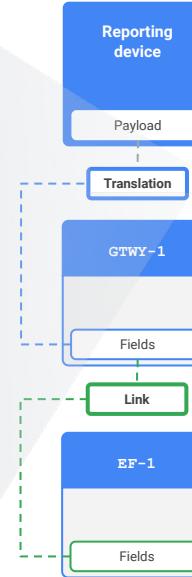
This link isn't configured correctly because it doesn't name the source entity. In this sample, the source entity is **GTWY-1** and the target entity is **EF-1**.

Try again

```
replace-with-guid-of-ef-1: #NOTE This is target device.  
code: EF-1  
type: FAN_SS_...  
links:  
  speed_frequency_sensor: speed_frequency_sensor_1  
  power_sensor: power_sensor_1  
...
```

```
replace-with-guid-of-gtwy-1: #NOTE This is target device.  
code: GTWY-1  
type: GATEWAYS/PASSTHROUGH  
translation:  
  speed_frequency_sensor_1:  
    present_value: points.ef_1_speed.present_value  
    units:  
      key: points.ef_1_speed.units  
      value:  
        hertz: 'hertz'  
  speed_frequency_sensor_2:  
    present_value: points.ef_2_speed.present_value  
    units:  
      key: points.ef_2_speed.units  
      value:  
        hertz: 'hertz'  
  power_sensor_1:  
    present_value: points.ef_1_kw.present_value  
    units:  
      key: points.ef_1_kw.units  
      value:  
        kilowatts: 'kw'  
  power_sensor_2:  
    present_value: points.ef_2_kw.present_value  
    units:  
      key: points.ef_2_kw.units  
      value:  
        kilowatts: 'kw'  
...
```

```
replace-with-guid-of-ef-1: #NOTE This is target device.  
code: EF-1  
type: FAN_SS_...  
links:  
  the-gtwy-guid:  
    speed_frequency_sensor_1: speed_frequency_sensor  
    power_sensor_1: power_sensor  
...
```



Back

Next

Hmm, that's not right! 🤔

This link isn't configured correctly because the fields aren't listed in the correct order in the source entity block. The first field should be from the target entity **EF-1**, and the second field should be from the source entity **GTWY-1**.

Try again

```
replace-with-guid-of-gtwy-1:  
  code: GTWY-1  
  type: GATEWAYS/PASSTHROUGH  
  translation:  
    speed_frequency_sensor_1:  
      present_value: points.ef_1_speed.present_value  
      units:  
        key: points.ef_1_speed.units  
        value:  
          hertz: 'hertz'  
    speed_frequency_sensor_2:  
      present_value: points.ef_2_speed.present_value  
      units:  
        key: points.ef_2_speed.units  
        value:  
          hertz: 'hertz'  
    power_sensor_1:  
      present_value: points.ef_1_kw.present_value  
      units:  
        key: points.ef_1_kw.units  
        value:  
          kilowatts: 'kw'  
    power_sensor_2:  
      present_value: points.ef_2_kw.present_value  
      units:  
        key: points.ef_2_kw.units  
        value:  
          kilowatts: 'kw'  
...  
...
```

```
replace-with-guid-of-ef-1: #NOTE This is target device.  
  code: EF-1  
  type: FAN_SS_...  
  links:  
    speed_frequency_sensor: speed_frequency_sensor_1  
    power_sensor: power_sensor_1  
...  
...
```

```
replace-with-guid-of-ef-1: #NOTE This is target device.  
  code: EF-1  
  type: FAN_SS_...  
  links:  
    the-gtwy-guid:  
      speed_frequency_sensor_1: speed_frequency_sensor  
      power_sensor_1: power_sensor  
...  
...
```

```
replace-with-guid-of-ef-1: #NOTE This is target device.  
  code: EF-1  
  type: FAN_SS_...  
  links:  
    the-gtwy-guid:  
      speed_frequency_sensor: speed_frequency_sensor_1  
      power_sensor: power_sensor_1  
...  
...
```

Back

Next

That's right! 🎉

This link is configured correctly, because:

- It names the target entity **EF-1**.
- It has a canonical entity type **FAN_SS_...**
- It names the source entity **GTWY-1**.
- It lists the linked fields in the correct order:
 - First field is from the target entity.
 - Second field is from the source entity.

```
replace-with-guid-of-gtwy-1:  
  code: GTWY-1  
  type: GATEWAYS/PASSTHROUGH  
  translation:  
    speed_frequency_sensor_1:  
      present_value: points.ef_1_speed.present_value  
      units:  
        key: points.ef_1_speed.units  
        value:  
          hertz: 'hertz'  
    speed_frequency_sensor_2:  
      present_value: points.ef_2_speed.present_value  
      units:  
        key: points.ef_2_speed.units  
        value:  
          hertz: 'hertz'  
    power_sensor_1:  
      present_value: points.ef_1_kw.present_value  
      units:  
        key: points.ef_1_kw.units  
        value:  
          kilowatts: 'kw'  
    power_sensor_2:  
      present_value: points.ef_2_kw.present_value  
      units:  
        key: points.ef_2_kw.units  
        value:  
          kilowatts: 'kw'  
...  
...
```

```
replace-with-guid-of-ef-1: #NOTE This is target device.  
  code: EF-1  
  type: FAN_SS_...  
  links:  
    speed_frequency_sensor: speed_frequency_sensor_1  
    power_sensor: power_sensor_1  
...  
...
```

```
replace-with-guid-of-ef-1: #NOTE This is target device.  
  code: EF-1  
  type: FAN_SS_...  
  links:  
    the-gtwy-guid:  
      speed_frequency_sensor_1: speed_frequency_sensor  
      power_sensor_1: power_sensor  
...  
...
```

```
replace-with-guid-of-ef-1: #NOTE This is target device.  
  code: EF-1  
  type: FAN_SS_...  
  links:  
    the-gtwy-guid:  
      speed_frequency_sensor: speed_frequency_sensor_1  
      power_sensor: power_sensor_1  
...  
...
```

[Back](#)

Click **Next** to move on and learn about mapping with passthrough entities.

[Next](#)

Mapping with a passthrough entity

A **passthrough entity** may be needed if a **reporting device** can't map one-to-one with a **logical entity**.

What's a passthrough entity?

A **passthrough entity** is a reporting entity that does nothing more than pass data from a network controller to logical entities.

Why are passthrough entities needed?

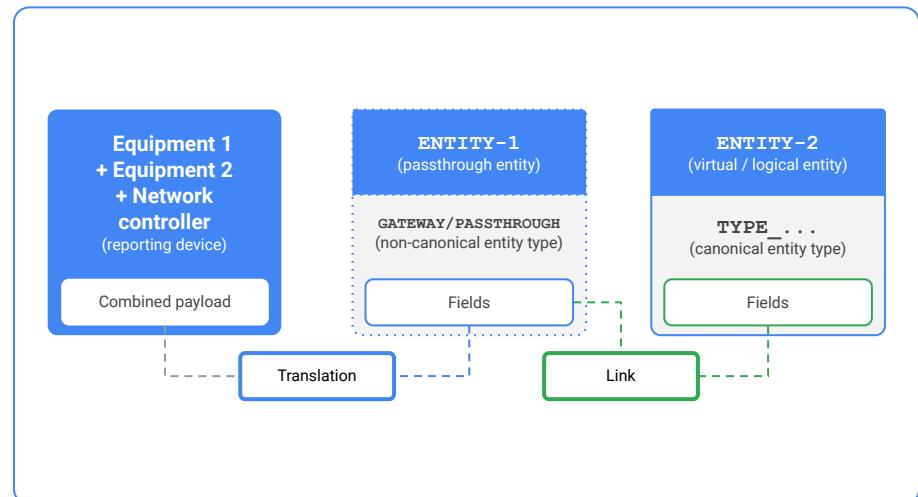
The only use case for a passthrough entity is when a network controller needs to be mapped. Network controllers send a combined payload for multiple devices, but there isn't a canonical type for them in the DBO.

In these instances, the network controller's combined payload needs to be translated to a passthrough entity, then linked to a virtual entity for each device in the combined payload.

What's a passthrough entity type?

Passthrough entities use the special entity type [GATEWAYS/PASSTHROUGH](#).

It's a non-canonical type with a unique feature allowing the passthrough entity to have any standard field defined on it for translations.



[Back](#)

Note: Passthrough entity types are reserved exclusively for the use of passthrough entities.

[Next](#)

Mapping with a passthrough entity (continued)

A **passthrough entity** does nothing more than pass data from a network controller to **logical entities**.

Example

Let's say you have two exhaust fans that you'd like to model. The fans have a network controller that sends their combined payload to the cloud, which makes it a reporting device. However, the cloud doesn't recognize the combined native payload.

Exhaust fan 1
+ Exhaust fan 2
+ Network controller
(reporting device)

Combined payload

Combined payload

```
{  
  "timestamp": "2021-10-18T09:52:43.000Z",  
  "points": [  
    {"ef_1_speed": {  
      "present_value": 60,  
      "units": "hertz"  
    }},  
    {"ef_2_speed": {  
      "present_value": 45,  
      "units": "hertz"  
    }},  
    {"ef_1_kw": {  
      "present_value": 12.77,  
      "units": "kw"  
    }},  
    {"ef_2_kw": {  
      "present_value": 6.753,  
      "units": "kw"  
    }}  
  ]  
}
```

Back

Click **Next** to inspect the passthrough entity in building config format.

Next

Mapping with a passthrough entity (continued)

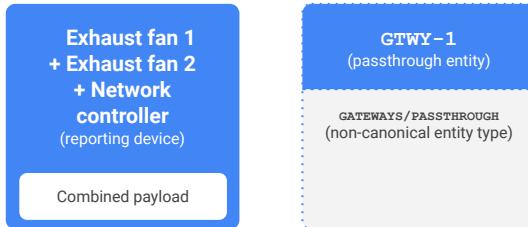
A **passthrough entity** does nothing more than pass data from a network controller to **logical entities**.

Example (continued)

The network controller doesn't have a canonical entity type. Instead, `GATEWAYS/PASSTHROUGH` is needed, which is a non-canonical entity type reserved exclusively for instances involving network controllers and passthrough entities.

We'll generate a GUID to properly identify the passthrough entity for the network controller, reference it with the code `GTWY-1`, and associate it with the special entity type `GATEWAYS/PASSTHROUGH`.

Now the network controller's combined payload can be translated.



Passthrough entity type

```
PASSTHROUGH:  
  id: "5942334369254342656"  
  description: "A device that provides  
  translations for virtual entities."  
  allow_undefined_fields: true
```

[Back](#)

Click [Next](#) to inspect the translation in building config format (human-readable).

[Next](#)

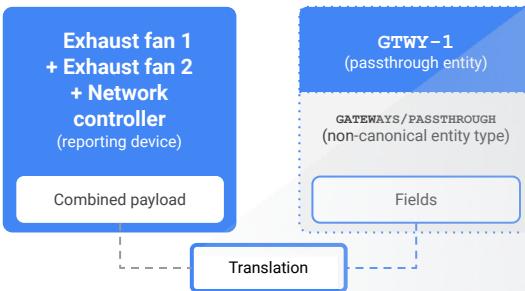
Mapping with a passthrough entity (continued)

A **passthrough entity** does nothing more than pass data from a network controller to **logical entities**.

Example (continued)

Using a translation, you can map the combined payload to the passthrough entity **GTWY-1** and any required standard fields.

Now the network controller is able to send its translated payload, and the cloud can recognize it. However, the exhaust fans aren't modeled as their own devices yet.



Translation 🕶️

```
replace-with-guid-of-gtwy-1:  
  code: GTWY-1  
  type: GATEWAYS/PASSTHROUGH#NOTE This type is just for gateways.  
  translation:  
    speed_frequency_sensor_1:  
      present_value: points.ef_1_speed.present_value  
      units:  
        key: points.ef_1_speed.units  
        value:  
          hertz: 'hertz'  
    speed_frequency_sensor_2:  
      present_value: points.ef_2_speed.present_value  
      units:  
        key: points.ef_2_speed.units  
        value:  
          hertz: 'hertz'  
    power_sensor_1:  
      present_value: points.ef_1_kw.present_value  
      units:  
        key: points.ef_1_kw.units  
        value:  
          kilowatts: 'kw'  
    power_sensor_2:  
      present_value: points.ef_2_kw.present_value  
      units:  
        key: points.ef_2_kw.units  
        value:  
          kilowatts: 'kw'  
...  
...
```

[Back](#)

Click **Next** to inspect the links in building config format (human-readable).

[Next](#)

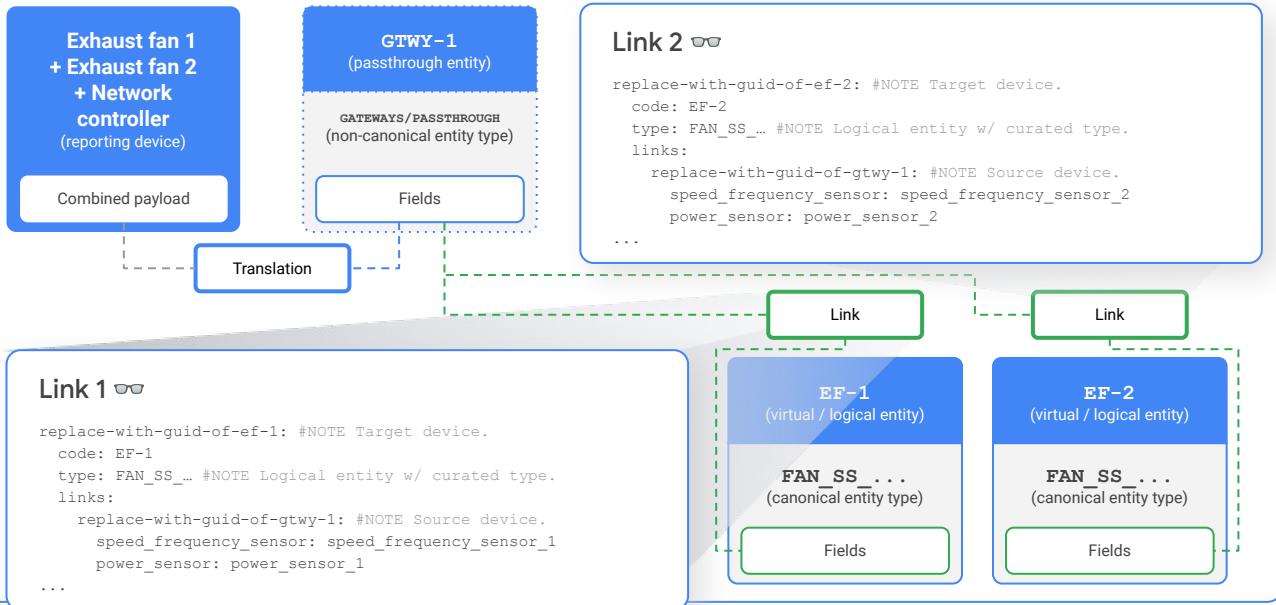
Mapping with a passthrough entity (continued)

A **passthrough entity** does nothing more than pass data from a network controller to **logical entities**.

Example (continued)

The exhaust fans are types of `FAN_SS_...` and canonical entity types. Since we care to model both of the fans, we'll generate GUIDs to properly identify these logical entities and reference them with the codes `EF-1` and `EF-2`.

Using two links, you can map the fields of `GTWY-1` to the fields of `EF-1` and `EF-2`. Now the cloud recognizes the combined native payload of the two exhaust fans.



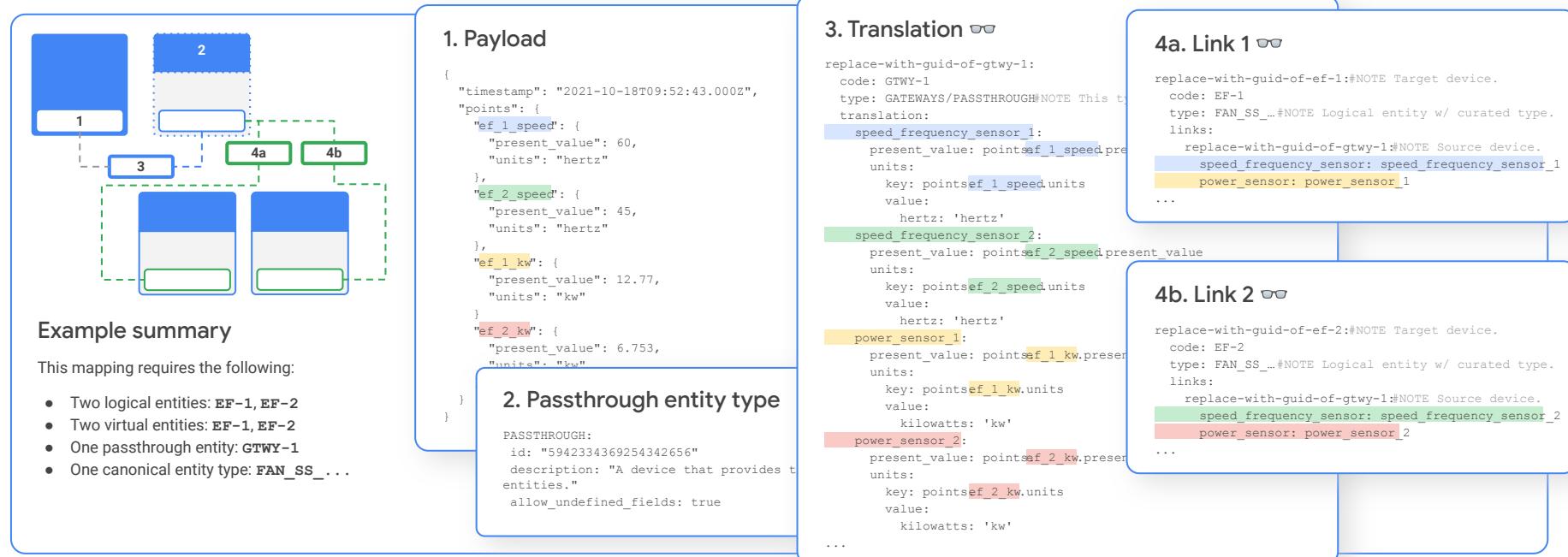
[Back](#)

Click [Next](#) for a summary of this example.

[Next](#)

Mapping with a passthrough entity (continued)

A **passthrough entity** does nothing more than pass data from a network controller to **logical entities**.



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, mappings should always be encoded in the correct building config format using an actual GUID.

Next

Lesson 7

Knowledge check 3



[Back](#)

Google

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

- The next slides will have one scenario with six questions to think about.
- Answer each question on your own and select the question to check your answer.
- After this knowledge check, you'll wrap up Lesson 7.

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

[Next](#)

Knowledge check 3

On the roof of a building, there are:

- Five identical, standard AHUs. Each AHU sends its own payload to the cloud through their individual onboard controllers
- Two identical, standard fans. Their payload is combined and sent to the cloud through a network controller.
- Three identical, standard heat pumps. Their payload is combined and sent to the cloud through another network controller.

Think about each question below. Select the question to check your answer.

How many **entities** are required to model this scenario?

How many **logical entities** are required to model this scenario?

How many **canonical entity types** are required to model this scenario?

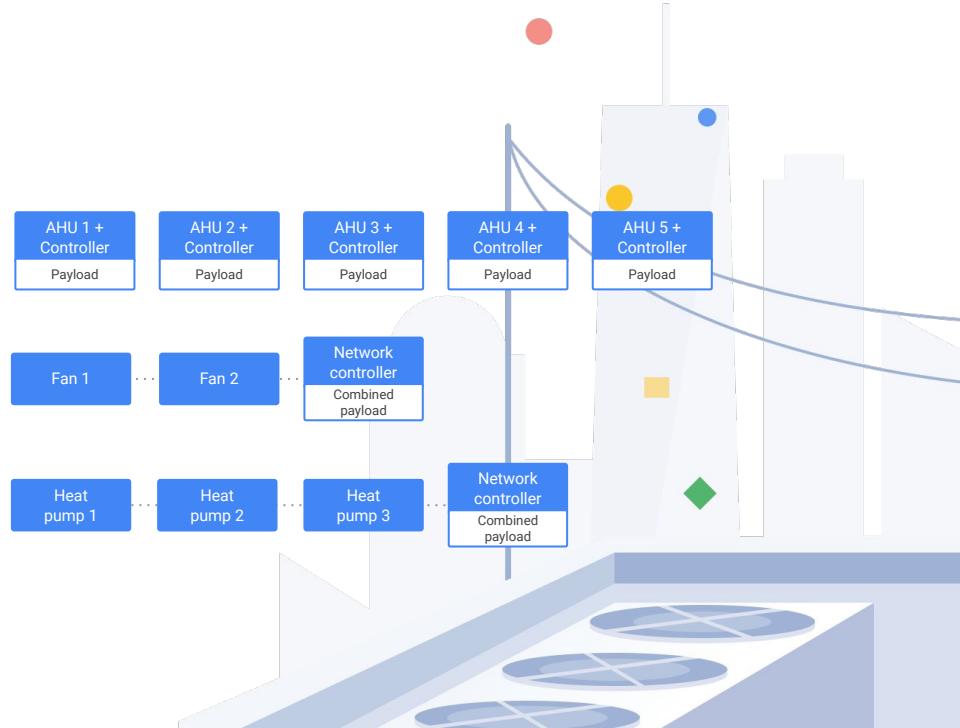
How many **reporting entities** are required to model this scenario?

How many **passthrough entities** are required to model this scenario?

How many **virtual entities** are required to model this scenario?

[Back](#)

[Next](#)



Knowledge check 3

On the roof of a building, there are:

- Five identical, standard AHUs. Each AHU sends its own payload to the cloud through their individual onboard controllers
- Two identical, standard fans. Their payload is combined and sent to the cloud through a network controller.
- Three identical, standard heat pumps. Their payload is combined and sent to the cloud through another network controller.

Think about each question below. Select the question to check your answer.

How many **entities** are required to model this scenario?

How many **logical entities** are required to model this scenario?

How many **canonical entity types** are required to model this scenario?

How many **reporting entities** are required to model this scenario?

How many **passthrough entities** are required to model this scenario?

How many **virtual entities** are required to model this scenario?

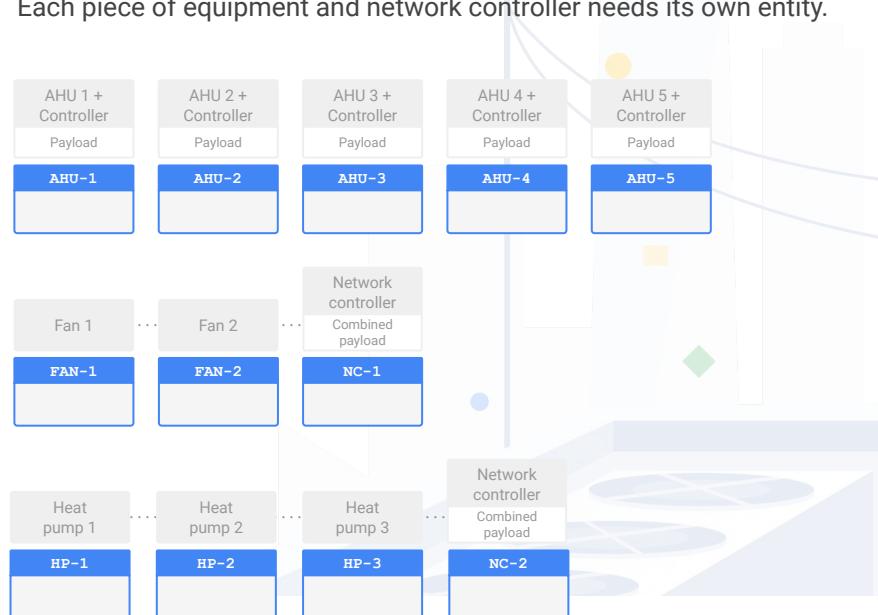
[Back](#)

[Next](#)

Check your answer!



A total of **twelve entities** are needed to model this scenario. Each piece of equipment and network controller needs its own entity.



Knowledge check 3

On the roof of a building, there are:

- Five identical, standard AHUs. Each AHU sends its own payload to the cloud through their individual onboard controllers
- Two identical, standard fans. Their payload is combined and sent to the cloud through a network controller.
- Three identical, standard heat pumps. Their payload is combined and sent to the cloud through another network controller.

Think about each question below. Select the question to check your answer.

How many **entities** are required to model this scenario?

How many **logical entities** are required to model this scenario?

How many **canonical entity types** are required to model this scenario?

How many **reporting entities** are required to model this scenario?

How many **passthrough entities** are required to model this scenario?

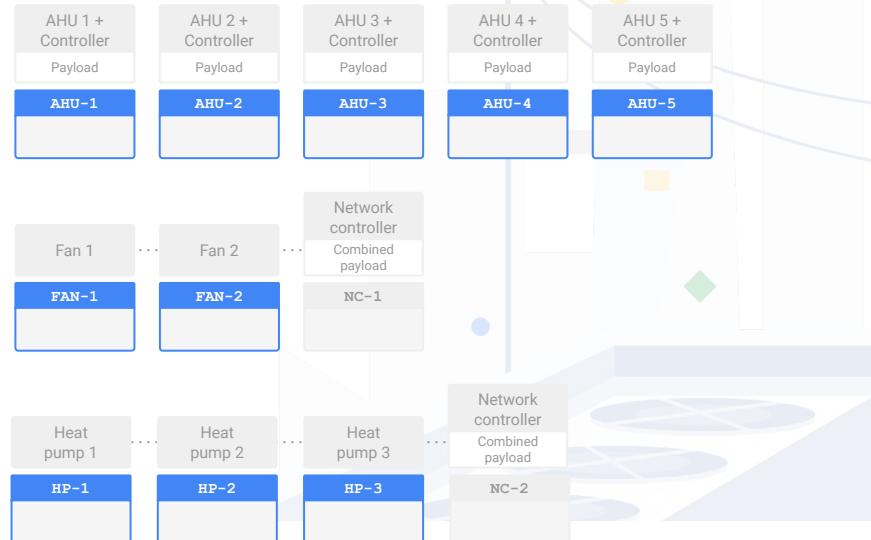
How many **virtual entities** are required to model this scenario?

[Back](#)

Check your answer!



Ten logical entities are needed: one for each device that needs to be modeled. A logical entity is also called a canonical entity. It maps one-to-one with a canonical entity type.



[Next](#)

Knowledge check 3

On the roof of a building, there are:

- Five identical, standard AHUs. Each AHU sends its own payload to the cloud through their individual onboard controllers
- Two identical, standard fans. Their payload is combined and sent to the cloud through a network controller.
- Three identical, standard heat pumps. Their payload is combined and sent to the cloud through another network controller.

Think about each question below. Select the question to check your answer.

How many **entities** are required to model this scenario?

How many **logical entities** are required to model this scenario?

How many **canonical entity types** are required to model this scenario?

How many **reporting entities** are required to model this scenario?

How many **passthrough entities** are required to model this scenario?

How many **virtual entities** are required to model this scenario?

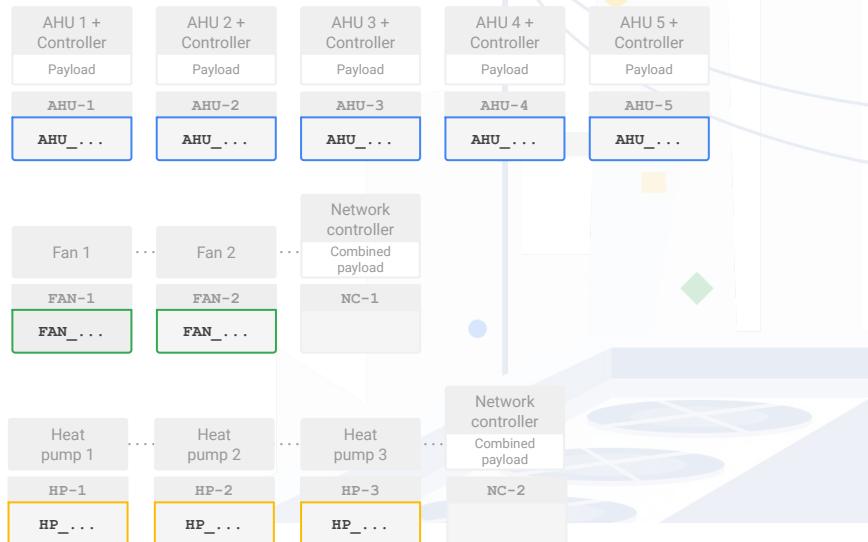
[Back](#)

[Next](#)

Check your answer!



Three canonical entity types are needed. One for the AHUs, one for the fans, and one for the heat pumps. Network controllers don't have a canonical type.



Knowledge check 3

On the roof of a building, there are:

- Five identical, standard AHUs. Each AHU sends its own payload to the cloud through their individual onboard controllers
- Two identical, standard fans. Their payload is combined and sent to the cloud through a network controller.
- Three identical, standard heat pumps. Their payload is combined and sent to the cloud through another network controller.

Think about each question below. Select the question to check your answer.

How many **entities** are required to model this scenario?

How many **logical entities** are required to model this scenario?

How many **canonical entity types** are required to model this scenario?

How many **reporting entities** are required to model this scenario?

How many **passthrough entities** are required to model this scenario?

How many **virtual entities** are required to model this scenario?

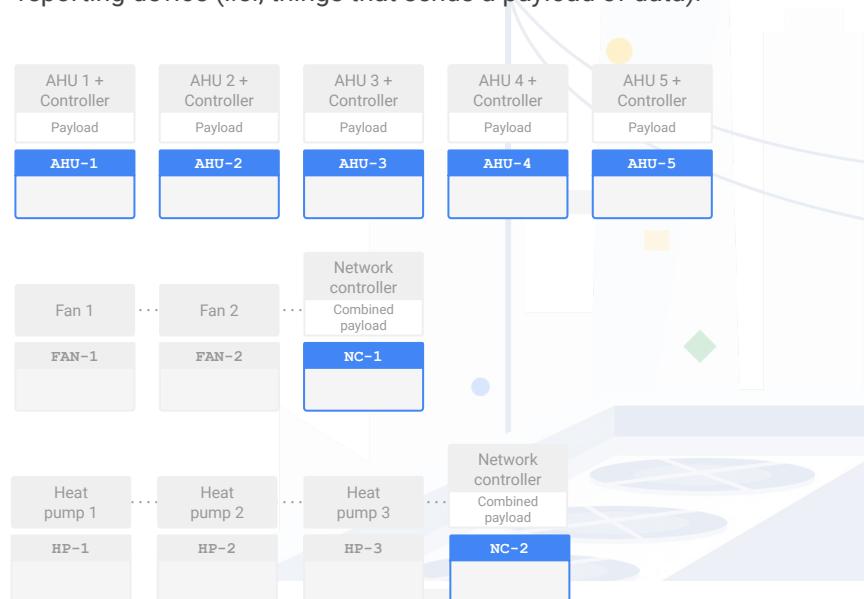
[Back](#)

[Next](#)

Check your answer!



Seven reporting entities are needed. One reporting entity for each reporting device (i.e., things that send a payload of data).



Knowledge check 3

On the roof of a building, there are:

- Five identical, standard AHUs. Each AHU sends its own payload to the cloud through their individual onboard controllers
- Two identical, standard fans. Their payload is combined and sent to the cloud through a network controller.
- Three identical, standard heat pumps. Their payload is combined and sent to the cloud through another network controller.

Think about each question below. Select the question to check your answer.

How many **entities** are required to model this scenario?

How many **logical entities** are required to model this scenario?

How many **canonical entity types** are required to model this scenario?

How many **reporting entities** are required to model this scenario?

How many **passthrough entities** are required to model this scenario?

How many **virtual entities** are required to model this scenario?

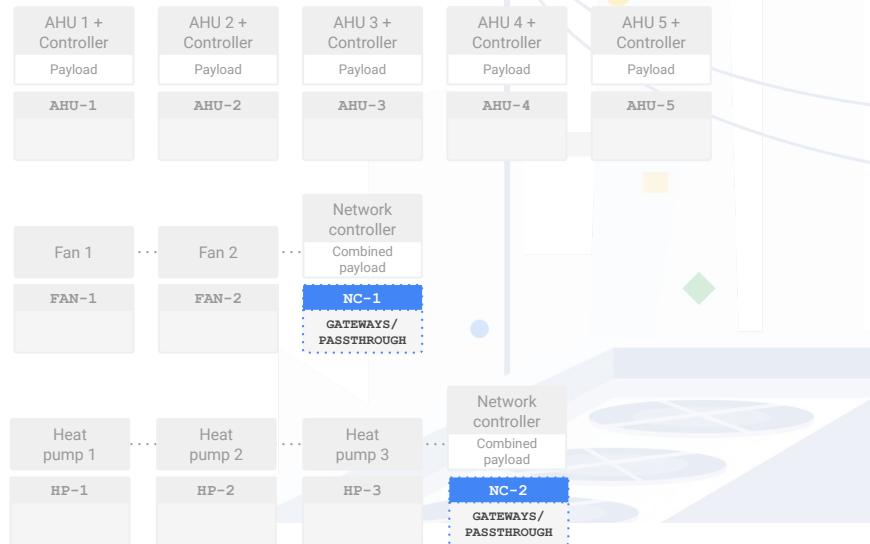
[Back](#)

[Next](#)

Check your answer!



Two passthrough entities are needed. The two network controllers can't map one-to-one with a logical entity. The passthrough entities will use the entity type **GATEWAYS/PASSTHROUGH**.



Knowledge check 3

On the roof of a building, there are:

- Five identical, standard AHUs. Each AHU sends its own payload to the cloud through their individual onboard controllers
- Two identical, standard fans. Their payload is combined and sent to the cloud through a network controller.
- Three identical, standard heat pumps. Their payload is combined and sent to the cloud through another network controller.

Think about each question below. Select the question to check your answer.

How many **entities** are required to model this scenario?

How many **logical entities** are required to model this scenario?

How many **canonical entity types** are required to model this scenario?

How many **reporting entities** are required to model this scenario?

How many **passthrough entities** are required to model this scenario?

How many **virtual entities** are required to model this scenario?

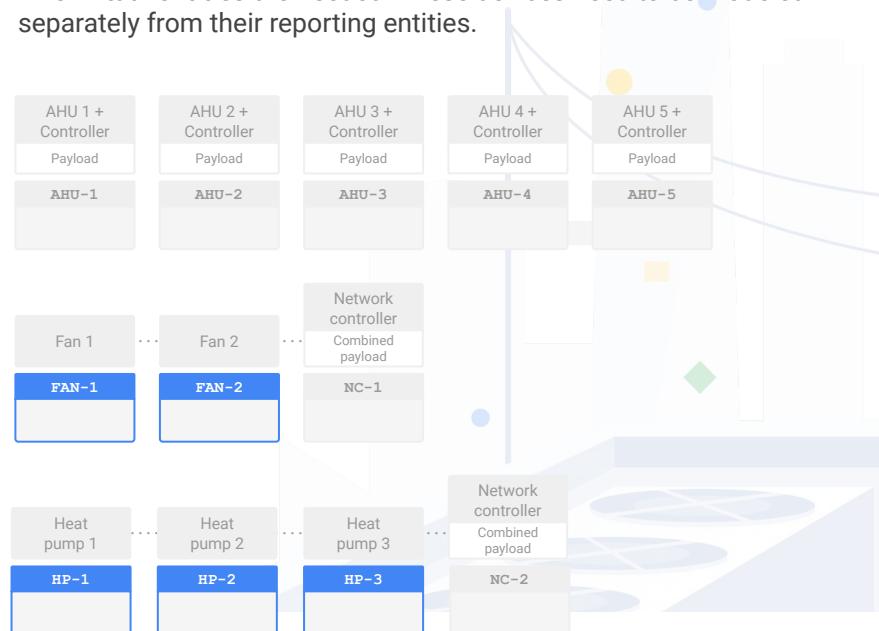
[Back](#)

[Next](#)

Check your answer!



Five virtual entities are needed. These devices need to be modeled separately from their reporting entities.



Lesson 7 summary

Let's review what you learned about:

- Mappings and logical devices
- Translations, reporting devices, reporting entities, and point mapping
- Links, virtual entities, and passthrough entities

Now you should be able to:

- Describe the purpose of mapping.
- Recognize the correct configuration of a translation.
- Recognize the correct configuration of a link.
- Describe the need for point mapping.
- Understand the dependencies between links and translations for data modeling.
- Determine when reporting entities, virtual entities, and passthrough entities are needed.



[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, mappings should always be encoded in the correct building config format using an actual GUID.

[Next](#)

You completed Lesson 7!

Now's a great time to take a quick break before starting Lesson 8.

Ready for Lesson 8?

Let's go!

Back

Helpful resources

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

- [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.