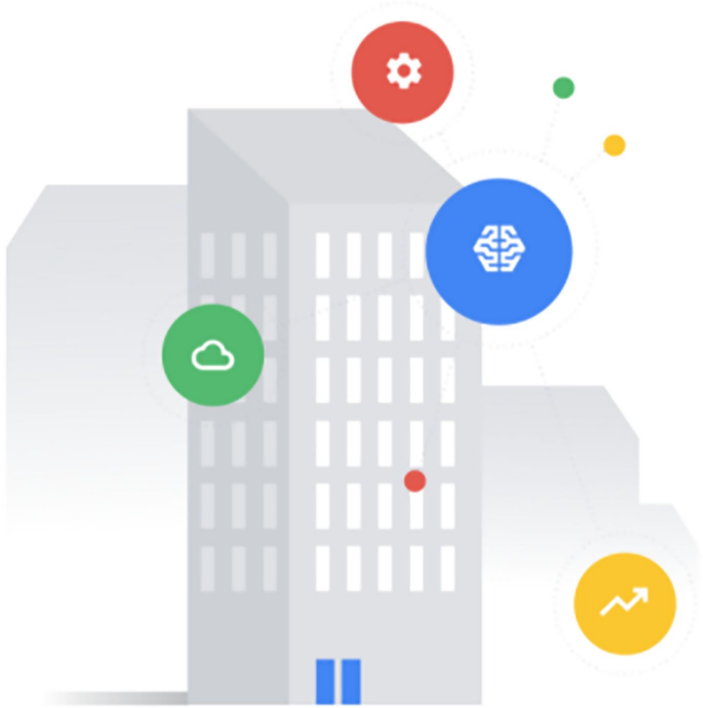
Google

Module 1 | **Lesson 4**

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!

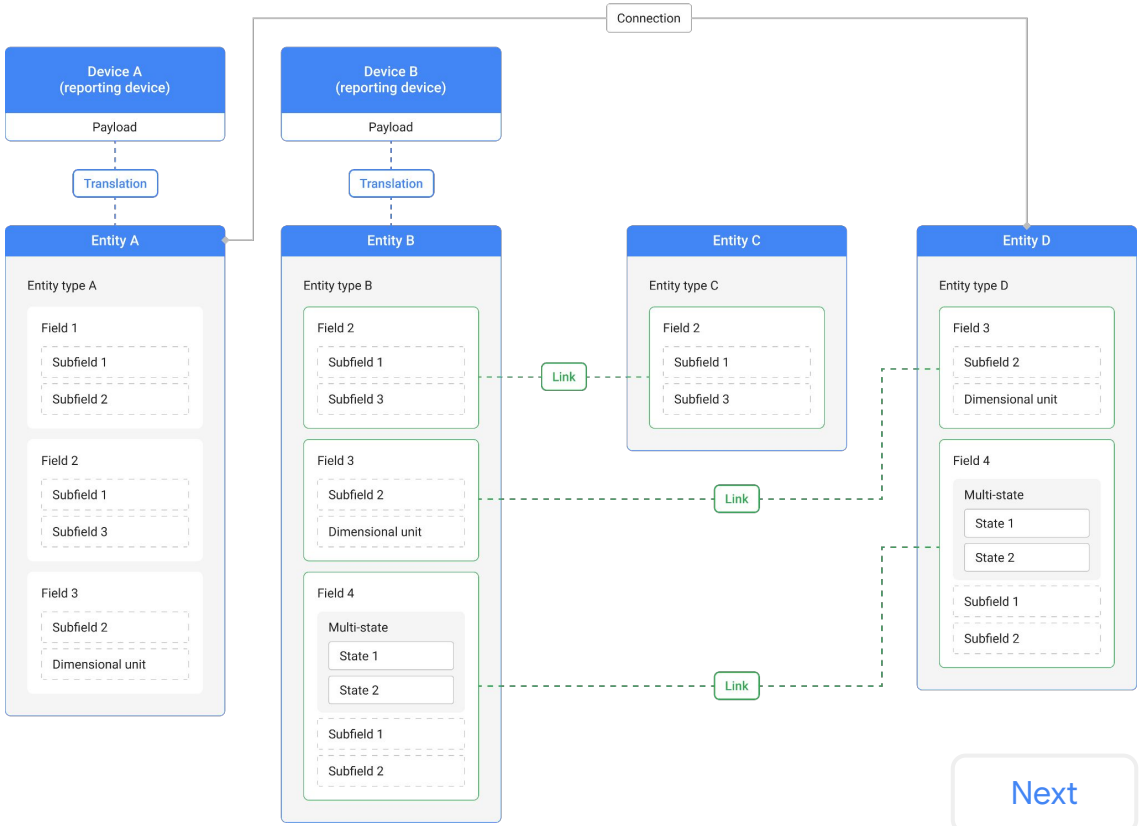
Conceptual model revisited

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, abstract modeling concepts are used to describe the properties of an entity. Abstract concepts include:

- Subfields
- Fields
- States and multi-states
- Entity types

Do you see these concepts in the diagram?

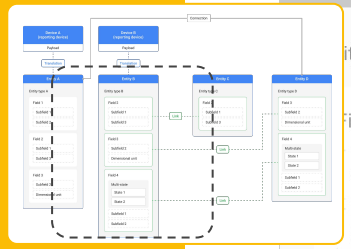


Back

Next

Lesson 4

Fields



What you'll learn about:

- Fields
- Field construction syntax
- Field constraints
- Field enumeration

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

- Describe the concept of a field.
- Identify a field in source code.
- Construct a field using the correct syntax.
- Recognize when enumeration is applied to a field.

Back

Google

Entity A

ity type A

ield 1

Subfield 1

Subfield 2

Field 2

Subfield 1

Subfield 3

Field 3

Subfield 2

Dimensional unit

Entity B

Entity type B

Field 2

Subfield 1

Subfield 3

Field 3

Subfield 2

Dimensional unit

Field 4

Multi-state

State 1

State 2

Subfield 1

Subfield 2

Link

Entity typ

Field 2

Subfi

Subfi

Next

What's a field?

A field is a grouping of subfields in a structured way to define a concept semantically.

Each field consists of a string of subfields. Fields are self-descriptive, and you should be able to read it naturally.

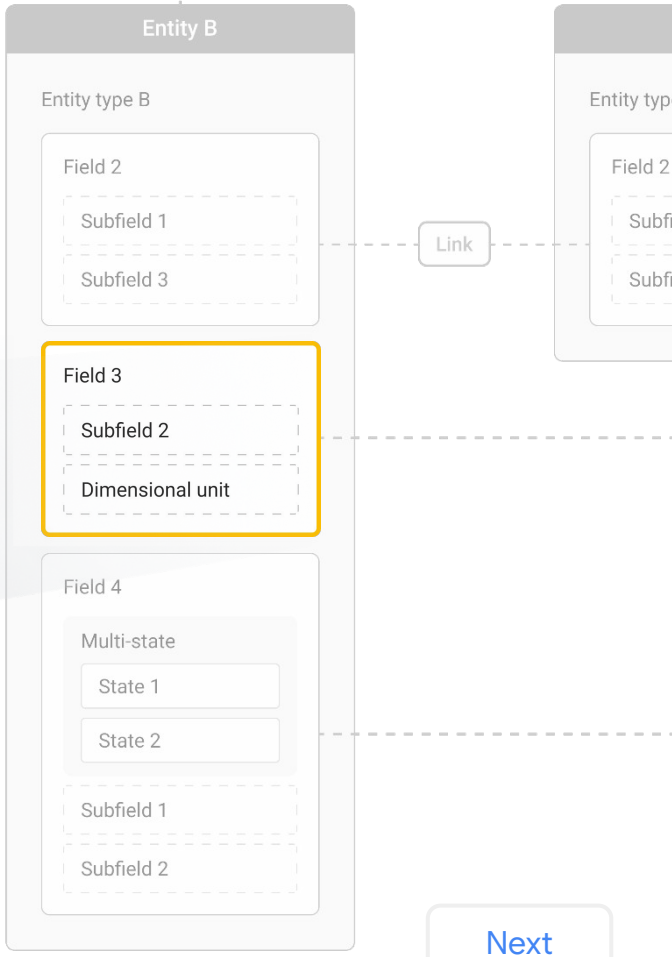
Examples

In the below examples, each individual word is a subfield and separated by underscores.

```
zone_air_temperature_sensor
```

```
supply_fan_run_command
```

See digitalbuildings / ... / resources / fields in the Digital Buildings Project GitHub repo for a list of all available telemetry and metadata fields.



Back

Next

Field construction syntax

Fields are constructed by combining a point type and at least one other subfield in a very specific order:

`(<Agg_Desc>_)?(<Agg>_)?(<descr>_)*(<component>_)?(<meas. descr>_)?(<meas>_)?<PointType>(_<num> )*`

Order of subfields

Subfields should be combined in this specific order:

1. **`(<Agg_Desc>_)?`** – Aggregation descriptor subfield; optional; can only be used once in conjunction with an aggregation subfield
2. **`(<Agg>_)?`** – Aggregation subfield; optional; can only be used once
3. **`(<descr>_)*`** – Descriptor subfield; optional; can only be used multiple times
4. **`(<component>_)?`** – Component subfield; optional; can only be used once
5. **`(<meas. descr>_)?`** – Measurement descriptor subfield; optional; can only be used once
6. **`(<meas>_)?`** – Measurement subfield; optional; can only be used once
7. **`<PointType>`** – Point type subfield; required; can only be used once

In practice, a combination of a point type and at least one other subfield will always be used to contextualize a field properly.

Example

Here's a sample field:

`max_discharge_air_temperature_setpoint`

It has combined the following subfields:

1. **`max`** – Aggregation subfield
2. **`discharge`** – Descriptor subfield
3. **`air`** – Descriptor subfield
4. **`temperature`** – Measurement subfield
5. **`setpoint`** – Point type subfield

The following subfields weren't used:

- Aggregation descriptor, because **`max`** isn't a time-related aggregation.
- Component, because **`air_temperature`** doesn't apply to a specific subcomponent of the entity.
- Measurement descriptor, because we can safely assume this **`temperature`** field describes a dry-bulb temperature per precedent. A wet-bulb temperature would need a measurement descriptor.

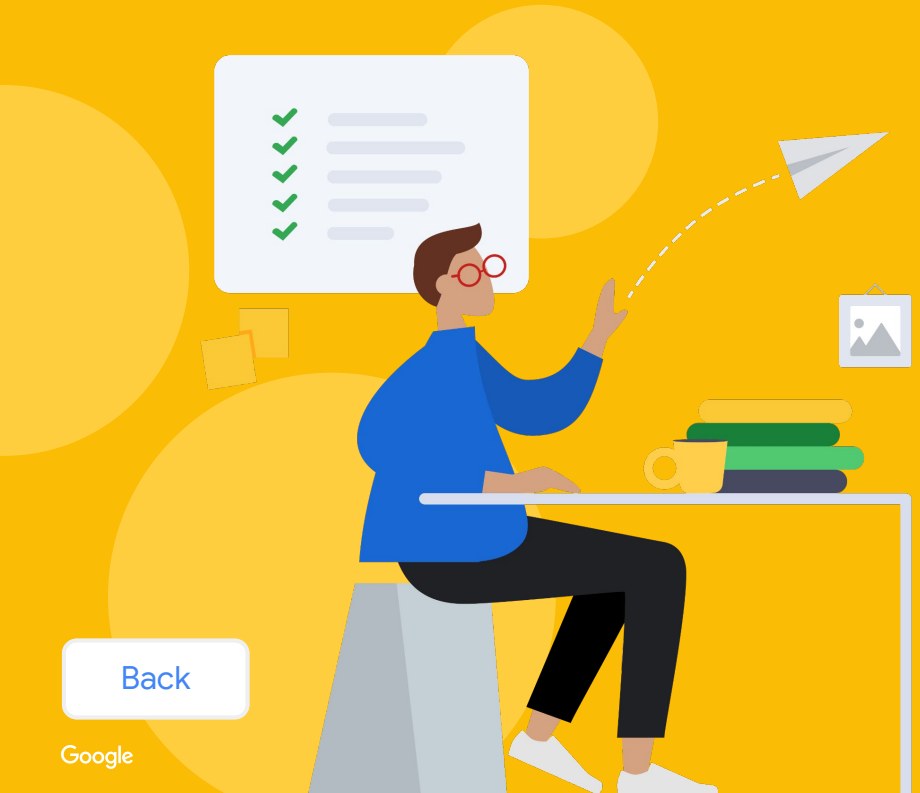
Back

Note: Field construction syntax is written in regular expression syntax! Need help with regex?
Visit RegexOne for more info or regex101 for a tool to test patterns.

Next

Lesson 4

Practice



Let’s take a moment to try to construct a field.

- The next slide will have several subfields that need to be grouped into a field.
- Recall the proper field syntax and select the correct order of subfields.

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

Practice

To the right are several subfields that are needed to construct a field.

`?_?_?_?_?`

Which subfield should come first?

Select the best answer from the options listed below.

- `sensor` - Point type
- `water` - Descriptor
- `secondary` - Descriptor
- `temperature` - Measurement
- `return` - Descriptor

- `sensor` - Point type subfield
- `water` - Descriptor subfield
- `secondary` - Descriptor subfield
- `temperature` - Measurement subfield
- `return` - Descriptor subfield

Back

Note: More than one option may apply. If so, select only one option.

Next

Google

Practice

To the right are several subfields that are needed to construct a field.

`sensor_?_?_?_?`

Which subfield should come first?

Select the best answer from the options listed below.

- **`sensor`** - Point type
- `water` - Descriptor
- `secondary` - Descriptor
- `temperature` - Measurement
- `return` - Descriptor

Hmm, that's not right! 🤔

Remember, a field shouldn't begin with a point type subfield.

Try again

Back

Next

Practice

To the right are several subfields that are needed to construct a field.

`secondary_return_water_?_?`

Which subfield should come first?

Select the best answer from the options listed below.

`sensor` - Point type

`water` - Descriptor

`secondary` - Descriptor

`temperature` - Measurement

`return` - Descriptor

Good job! 🎉

Any of the descriptor subfields would work here, since the order of descriptors isn't strictly enforced.

Remember, descriptor subfields are used to specify the exact function of the field within the context of the entity. Descriptors can be used multiple times in a field, but you should use as few descriptors as possible.

Although the order of descriptors is not strictly enforced, you should always rely on readability and precedent to determine ordering. In this case, **`secondary_return_water_...`** is preferred because it is more intuitive to read than **`return_secondary_water_...`**, **`water_return_secondary_...`**. Plus, there's already a precedent set in the ontology for it, which should be adhered to as much as possible.

Back

Next

Practice

To the right are several subfields that are needed to construct a field.

`temperature_?_?_?_?`

Which subfield should come first?

Select the best answer from the options listed below.

- `sensor` - Point type
- `water` - Descriptor
- `secondary` - Descriptor
- **`temperature` - Measurement**
- `return` - Descriptor

Hmm, that's not right! 🤔

Remember, a measurement subfield shouldn't come before a descriptor.

Try again

Back

Next

Practice

To the right are several subfields that are needed to construct a field.

`secondary_return_water_?_?`

Which subfield should come next?

Select the best answer from the options listed below.

- **`sensor`** - Point type
- **`temperature`** - Measurement

- **`sensor`** - Point type subfield
- **`water`** - Descriptor subfield
- **`secondary`** - Descriptor subfield
- **`temperature`** - Measurement subfield
- **`return`** - Descriptor subfield

Back

Next

Google

Practice

To the right are several subfields that are needed to construct a field.

`secondary_return_water_sensor_?`

Which subfield should come next?

Select the best answer from the options listed below.

- **`sensor`** - Point type
- `temperature` - Measurement

Hmm, that's not right! 🤔

Remember, a point type subfield shouldn't come before a measurement subfield.

Try again

Back

Next

Practice

To the right are several subfields that are needed to construct a field.

`secondary_return_water_temperature_?`

Which subfield should come next?

Select the best answer from the options listed below.

`sensor` - Point type

`temperature` - Measurement

Good job! 🎉

The measurement subfield should come before the point type subfield.

Remember, the measurement subfield serves a few purposes like identifying the type of measurement being performed and the field's dimensional unit.

Back

Next

Practice

To the right are several subfields that are needed to construct a field.

`secondary_return_water_temperature?`

There's only one subfield left.

Select the final option.

`sensor` - Point type

`sensor` - Point type subfield

`water` - Descriptor subfield

`secondary` - Descriptor subfield

`temperature` - Measurement subfield

`return` - Descriptor subfield

Back

Next

Google

Practice

To the right are several subfields that are needed to construct a field.

`secondary_return_water_temperature_sensor`

There's only one subfield left.

Select the final option.

`sensor` - Point type

You did it! 🎉

The point type subfield should always be the last subfield.

Remember, point types are the only subfield that's required for field construction since it identifies a point's function. However, it must be combined with at least one other subfield to contextualize a field.

In this practice exercise, the aggregation, component, and measurement descriptor subfields weren't used because they're optional subfields.

Back

Next

Field constraints

Remember the following constraints when working with fields.

Fields should always:

- Use a minimum of two subfields.
- Follow the construction syntax to combine subfields in the correct order.
- Use a measurement subfield for numeric values unless the point type subfield is `count` or `counter`.
- Use the minimum number of descriptor subfields needed to uniquely define the field within the context of your desired entity type.

Fields should normally:

- Be defined in the global namespace.
- Be as specific as the context dictates.

Fields should never:

- Use the same subfield more than once.
- Have the same set of subfields as another field with different ordering.
- Use an excessive amount of descriptors.
- Be identical to another field in a different namespace.

Back

Note: You'll learn about namespaces in Lesson 9 of this module.

Next

Field enumeration

The same field can be used more than once:

`(<Agg_Desc>_)?(<Agg>_)?(<descr>_)*(<component>_)?(<meas. descr>_)?(<meas>_)?<PointType> (_<num> )*`

Sometimes, an entity may be made of several of the same parts with identical functions. If this occurs, the field for each part needs to be numbered to differentiate between them.

Use case

A device has two zone temperature sensors that function identically.

To distinguish between the two sensors, one sensor uses the field **`<field>_1`**, and the other sensor uses **`<field>_2`**.

Subgrouping

Subgrouping is allowed and indicated by using multiple increments of enumeration: **`<field>_1_2`**, **`<field>_1_3`**, and so on.

Example

Here's an AHU that requires field enumeration.

```
AHU_US_MTV_2171_3:
  id: "11391598658907537408"
  description: "Non-standard type for 2171"
  implements:
  - AHU_BYPSSPC2X_DX2SC_ECON_HT2SC_SFSS
  - INCOMPLETE
  uses:
  - supply_air_static_pressure_sensor1
  - supply_air_static_pressure_sensor2
```

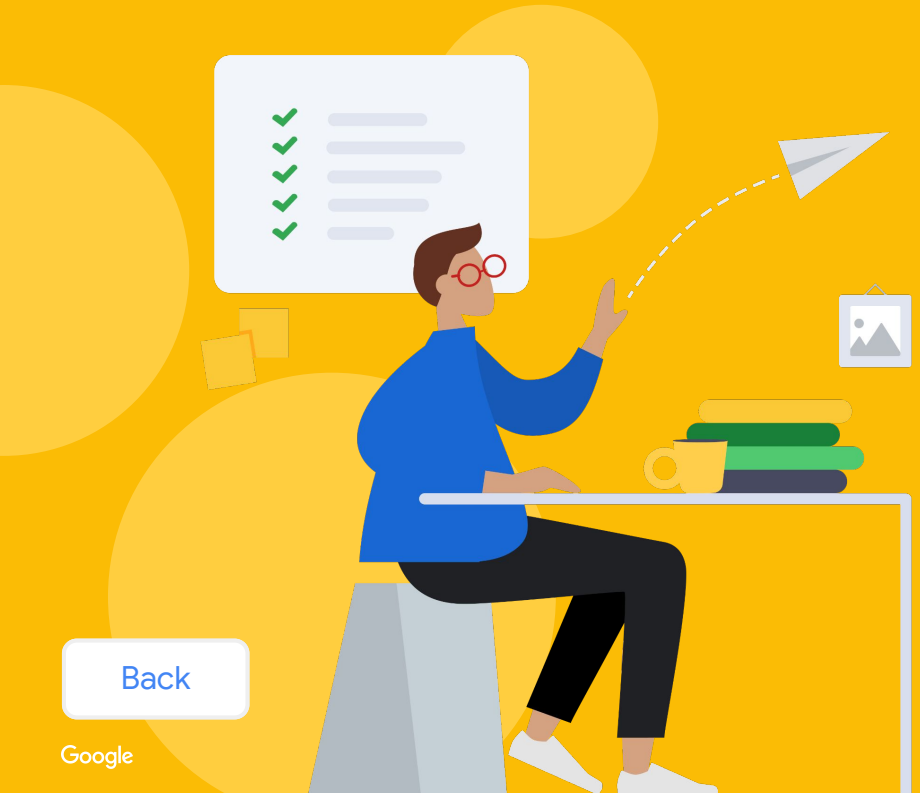
Back

Next

Google

Lesson 4

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 field is a grouping of _____ in a structured way to define concepts semantically.

Fill in the blank.

Select the best answer from the options listed below.

- points
- tags
- subfields
- components

Back

Next

Google

Knowledge check 1

A field is a grouping of _____ in a structured way to define concepts semantically.

Fill in the blank.

Select the best answer from the options listed below.

points

tags

subfields

components

Hmm, that's not right! 🤔

Try again

Back

Next

Google

Knowledge check 1

A field is a grouping of _____ in a structured way to define concepts semantically.

Fill in the blank.

Select the best answer from the options listed below.

points

tags

subfields

components

Hmm, that's not right! 🤔

Try again

Back

Next

Google

Knowledge check 1

A field is a grouping of _____ in a structured way to define concepts semantically.

Fill in the blank.

Select the best answer from the options listed below.

points

tags

subfields

components

That's right! 🎉

Fields consist of a string of subfields in very particular order.

Brevity is key with fields, so only the minimum number of subfields should be used to uniquely define a field.

Examples

```
zone_air_temperature_sensor
supply_fan_run_command
```

Back

Next

Google

Knowledge check 1

A field is a grouping of _____ in a structured way to define concepts semantically.

Fill in the blank.

Select the best answer from the options listed below.

points

tags

subfields

components

Hmm, that's not right! 🤔

Try again

Back

Next

Google

Knowledge check 2

Fields are constructed by grouping subfields in a very specific order.

What's the correct order of subfields?

Select the best answer from the options listed below.

- `<PointType>(_<num> )*(<descr>_)*(<component>_)?(<meas. descr>_)?(<meas>_)?(<Agg_Desc>_)?(<Agg>_)?`
- `(<Agg_Desc>_)?(<Agg>_)?(<descr>_)*(<component>_)?(<meas>_)?(<meas. descr>_)?<PointType>(_<num> )*`
- `(<Agg_Desc>_)?(<Agg>_)?(<meas. descr>_)?(<meas>_)?(<descr>_)*(<component>_)?<PointType>(_<num> )*`
- `(<Agg_Desc>_)?(<Agg>_)?(<descr>_)*(<component>_)?(<meas. descr>_)?(<meas>_)?<PointType>(_<num> )*`

Back

Next

Google

Knowledge check 2

Fields are constructed by grouping subfields in a very specific order.

Close... but not quite right! 🤔

Try again

What's the correct order of subfields?

Select the best answer from the options listed below.

`<PointType>(_<num> )*(<descr>_)*(<component>_)?(<meas. descr>_)?(<meas>_)?(<Agg_Desc>_)?(<Agg>_)?`

`(<Agg_Desc>_)?(<Agg>_)?(<descr>_)*(<component>_)?(<meas>_)?(<meas. descr>_)?<PointType>(_<num> )*`

`(<Agg_Desc>_)?(<Agg>_)?(<meas. descr>_)?(<meas>_)?(<descr>_)*(<component>_)?<PointType>(_<num> )*`

`(<Agg_Desc>_)?(<Agg>_)?(<descr>_)*(<component>_)?(<meas. descr>_)?(<meas>_)?<PointType>(_<num> )*`

Back

Next

Google

Knowledge check 2

Fields are constructed by grouping subfields in a very specific order.

Close... but not quite right! 🤔

Try again

What's the correct order of subfields?

Select the best answer from the options listed below.

<PointType>(_<num>)*(<descr>_)*(<component>_)?(<meas. descr>_)?(<meas>_)?(<Agg_Desc>_)?(<Agg>_)?

(<Agg_Desc>_)?(<Agg>_)?(<descr>_)*(<component>_)?(<meas>_)?(<meas. descr>_)?<PointType>(_<num>)*

(<Agg_Desc>_)?(<Agg>_)?(<meas. descr>_)?(<meas>_)?(<descr>_)*(<component>_)?<PointType>(_<num>)*

(<Agg_Desc>_)?(<Agg>_)?(<descr>_)*(<component>_)?(<meas. descr>_)?(<meas>_)?<PointType>(_<num>)*

Back

Next

Google

Knowledge check 2

Fields are constructed by grouping subfields in a very specific order.

Close... but not quite right! 🤔

Try again

What's the correct order of subfields?

Select the best answer from the options listed below.

`<PointType>(_<num> )*(<descr>_)*(<component>_)?(<meas. descr>_)?(<meas>_)?(<Agg_Desc>_)?(<Agg>_)?`

`(<Agg_Desc>_)?(<Agg>_)?(<descr>_)*(<component>_)?(<meas>_)?(<meas. descr>_)?<PointType>(_<num> )*`

`(<Agg_Desc>_)?(<Agg>_)?(<meas. descr>_)?(<meas>_)?(<descr>_)*(<component>_)?<PointType>(_<num> )*`

`(<Agg_Desc>_)?(<Agg>_)?(<descr>_)*(<component>_)?(<meas. descr>_)?(<meas>_)?<PointType>(_<num> )*`

Back

Next

Google

Knowledge check 2

Fields are constructed by grouping subfields in a very specific order.

What's the correct order of subfields?

Select the best answer from the options listed below.

`<PointType>(_<num> )*(<descr>_)*(<component>_)?(<meas. descr>_)?(<meas>_)?(<Agg_Desc>_)?(<Agg>_)?`

`(<Agg_Desc>_)?(<Agg>_)?(<descr>_)*(<component>_)?(<meas>_)?(<meas. descr>_)?<PointType>(_<num> )*`

`(<Agg_Desc>_)?(<Agg>_)?(<meas. descr>_)?(<meas>_)?(<descr>_)*(<component>_)?<PointType>(_<num> )*`

`(<Agg_Desc>_)?(<Agg>_)?(<descr>_)*(<component>_)?(<meas. descr>_)?(<meas>_)?<PointType>(_<num> )*`

That's right! 🎉

It's important to follow this syntax when constructing a field.

In addition to the order of subfields, remember that most of them are optional and can only be used once:

1. **`(<Agg_Desc>_)?`** – Aggregation descriptor subfield
   Optional; can only be used once in conjunction with an aggregation.
2. **`(<Agg>_)?`** – Aggregation subfield
   Optional; can only be used once.
3. **`(<descr>_)*`** – Descriptor subfield
   Optional; can be used multiple times.
4. **`(<component>_)?`** – Component subfield
   Optional; can only be used once.
5. **`(<meas. descr>_)?`** – Measurement descriptor subfield
   Optional; can only be used once.
6. **`(<meas>_)?`** – Measurement subfield
   Optional; can only be used once.
7. **`<PointType>`** – Point type subfield
   Required; can only be used once.

Back

Next

Knowledge check 3

The same subfield can be used more than once.

`secondary_secondary_return_water_temperature_sensor`

Is this statement true or false?

Select the best answer from the options listed below.

True

False

Back

Next

Knowledge check 3

The same subfield can be used more than once.

`secondary_secondary_return_water_temperature_sensor`

Is this statement true or false?

Select the best answer from the options listed below.

True

False

Hmm, that's not right! 🤔

Subfields should never be used more than once in a field.
The second secondary subfield should be removed, like so:

`secondary_return_water_temperature_sensor`

Also, fields should never use the same set of subfields as another field with a different ordering. These wouldn't be valid:

`secondary_return_water_temperature_sensor`

`secondary_water_return_temperature_sensor`

A field, however, can be used more than once.

Sometimes, an entity may be made of several of the same parts with identical functions. When this occurs, then fields need to be reused and enumerated to differentiate between each part (e.g., **`<field>_1`** and **`<field>_2`**).

Back

Next

Google

Knowledge check 3

The same subfield can be used more than once.

`secondary_secondary_return_water_temperature_sensor`

Is this statement true or false?

Select the best answer from the options listed below.

True

False

That's right! 🎉

Subfields should never be used more than once in a field.
The second secondary subfield should be removed, like so:

`secondary_return_water_temperature_sensor`

Also, fields should never use the same set of subfields as another field with a different ordering. These wouldn't be valid:

`secondary_return_water_temperature_sensor`

`secondary_water_return_temperature_sensor`

A field, however, can be used more than once.

Sometimes, an entity may be made of several of the same parts with identical functions. When this occurs, then fields need to be reused and enumerated to differentiate between each part (e.g., `<field>_1` and `<field>_2`).

Back

Next

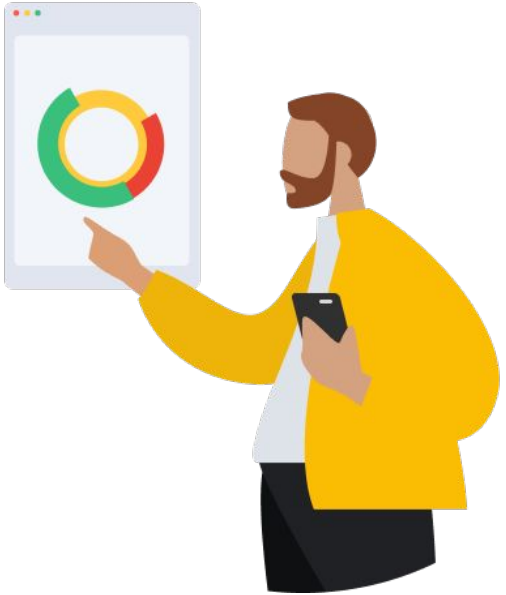
Lesson 4 summary

Let's review what you learned about:

- Fields
- Field construction syntax
- Field constraints
- Field enumeration

Now you should be able to:

- Describe the concept of a field.
- Identify a field in source code.
- Construct a field using the correct syntax.
- Recognize when enumeration is applied to a field.



Back

Next

Google

You completed Lesson 4!

Now's a great time to take a quick break before starting Lesson 5.

Ready for Lesson 5?

Let's go!

Back

Google

Helpful resources

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

- digitalbuildings / ontology / yaml / resources / fields
  Contains all of the available metadata and telemetry fields.
- digitalbuildings / ontology
  Contains the documentation and configuration files for the DBO.
- digitalbuildings / ontology / docs / ontology.md
  Provides an overview of the structure and principles of the ontology.
- digitalbuildings / ontology / docs / model.md
  Describes the conventions used in the DBO abstract model.
- Digital Buildings Project GitHub
  Contains source code, tooling, and documentation for the DBO.