



## 10: Entity Command





# Automatic Merge Fields

## From a Block or Rock Itself

```
1 {{ 'Lava' | Debug }}
```

**Lava Debug Info**

Below is a listing of available merge fields for this block. Find out more on Lava at [rockrms.com/lava](http://rockrms.com/lava).

Page Parameter

Current Person

Current Visitor

Campuses

Pagination

Linked Pages

Items

Item Tag List

Archive Summary

- RockVersion - 1.16.6.6
- CurrentPageUrl - /blog?Tag=TagTemplate
- ArchiveSummaryPageUrl - /blog?Year=YearTemplate&Month=MonthTemplate

Person

Global Attribute



# What if...

the record I need isn't in the list?



## Lava Debug Info

Below is a listing of available merge fields for this block. Find out more on Lava at [rockrms.com/lava](http://rockrms.com/lava).

### Page Parameter

### Current Person

### Current Visitor

### Campuses

### Pagination

### Linked Pages

### Items

### Item Tag List

### Archive Summary

- RockVersion - 1.16.6.6
- CurrentPageUrl - /blog?Tag=TagTemplate
- ArchiveSummaryPageUrl - /blog?Year=YearTemplate&Month=MonthTemplate

### Person

### Global Attribute



# What if...

the record I need isn't already provided?

If you need:

- A person:

```
1  {% assign person = 4 | PersonById %}
```

```
1  {% assign personAliasGuid = Item | Attribute:'Author','RawValue' %}  
2  {% assign person = personAliasGuid | PersonById %}
```





# What if...

the record I need isn't already provided?

If you need:

- A group:

```
1  {% assign group = 115 | GroupById %}
```

```
1  {% assign groupGuid = Item | Attribute:'Group','RawValue' %}
2  {% assign group = groupGuid | GroupById %}
```





# What if...

the record I need isn't already provided?

If you need:

- Another Entity Type:

There are no other “\_\_Byld” filters





# Commands

```
{% person where:'LastName == "Decker"' %}
```

- Familiar syntax
- Specific to Rock
  - (not available in other adaptions of Liquid)
- Custom filtering





# Entity Command

```
{% person where:'LastName == "Decker"' %}
```



# Entity Command

```
{% person where:'LastName == "Decker"' %}
```



# Entity Command

```
{% person where:'LastName == "Decker"' %}
```

```
{{ personItems }}
```



# Entity Command

```
{% person where:'LastName == "Decker"' %}
```

```
{{ personItems }}
```



# Entity Command

```
{% person where:'LastName == "Decker"' %}
```

```
{{ personItems }}
```

```
{% endperson %}
```



# Entity Command

```
1  {% person where:'LastName == "Decker"' %}  
2  
3  {% endperson %}
```

Works for all entity types

By default, this provides a variable named [entity\_type]Items, such as personItems.



# Entity Command

```
1  {% person where:'LastName == "Decker"' %}  
2    <pre>{{ personItems | ToJSON }}</pre>  
3  {% endperson %}
```



## Important to Know:

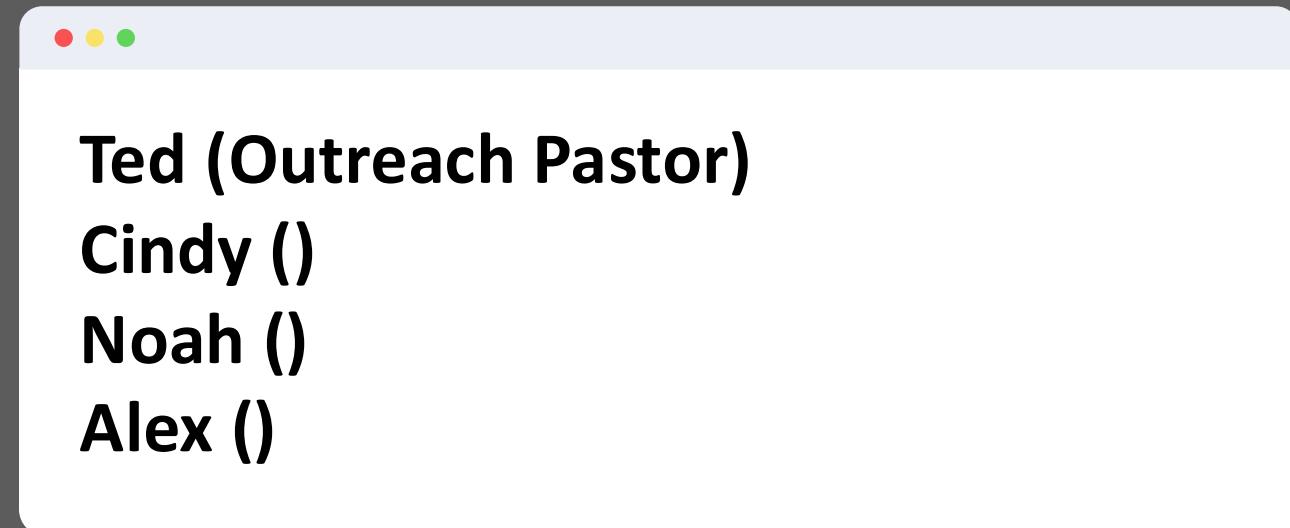
The elements in the array are already full entity records.

```
1  [  
2  {  
3    "Id": 4,  
4    "NickName": "Ted",  
5    "FirstName": "Theodore",  
6    "LastName": "Decker",  
7    "ConnectionStatusValueId": 65,  
8    "PhoneNumbers": [ ... ]  
9  },  
10 {  
11   "Id": 5,  
12   "NickName": "Cindy",  
13   "FirstName": "Cynthia",  
14   "LastName": "Decker",  
15   "ConnectionStatusValueId": 65,  
16   "PhoneNumbers": [ ... ]  
17 },  
18 {  
19   "Id": 6,  
20   "NickName": "Noah",  
21   "FirstName": "Noah",  
22   "LastName": "Decker",  
23   "ConnectionStatusValueId": 146,  
24   "PhoneNumbers": [ ... ]  
25 },  
26 {  
27   "Id": 7,  
28   "NickName": "Alex",  
29   "FirstName": "Alexis",  
30   "LastName": "Decker",  
31   "ConnectionStatusValueId": 146,  
32   "PhoneNumbers": [ ... ]  
33 }  
34 ]
```



# Entity Command

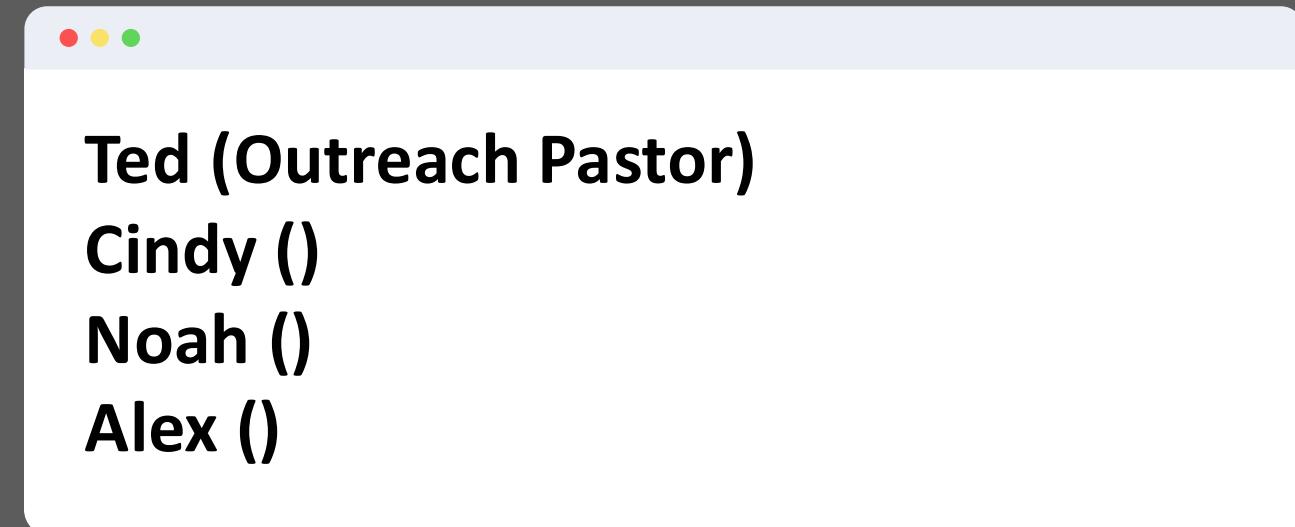
```
1  {% person where:'LastName == "Decker"' %}  
2    {% for individual in personItems %}  
3      <p>{{ individual.NickName }} ({{ individual | Attribute:'Position' }})</p>  
4    {% endfor %}  
5  {% endperson %}
```





# Entity Command

```
1  {% person where:'LastName == "Decker"' %}  
2    {% for person in personItems %}  
3      <p>{{ person.NickName }} ({{ person | Attribute:'Position' }})</p>  
4    {% endfor %}  
5  {% endperson %}
```





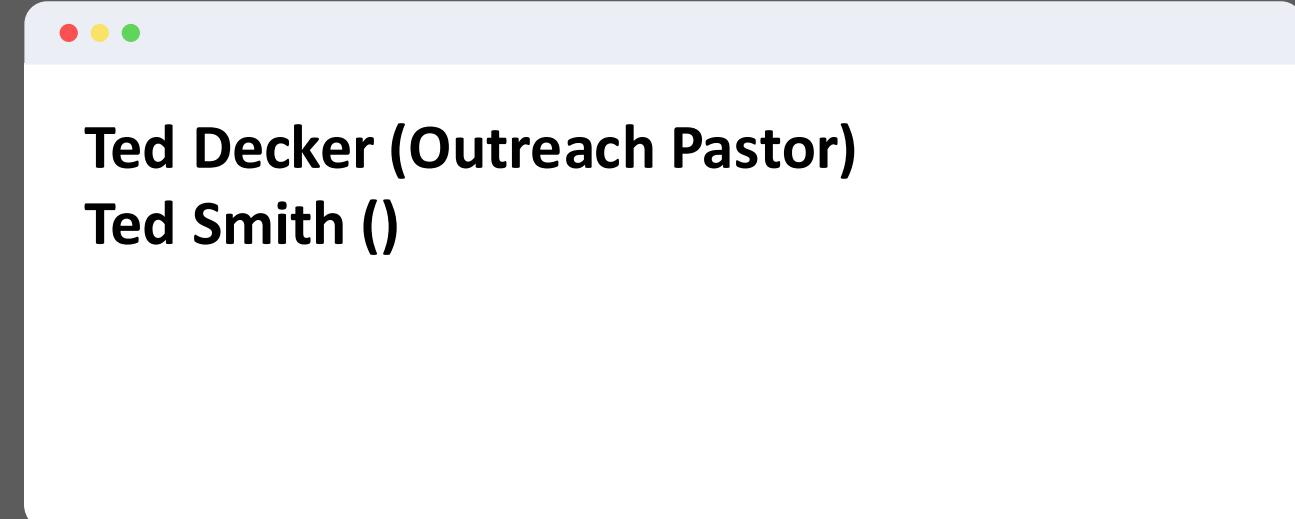
# Entity Command

```
1  {% person where:'NickName == "Ted"' %}  
2    {% for person in personItems %}  
3      <p>{{ person.FullName }} ({{ person | Attribute:'Position' }})</p>  
4    {% endfor %}  
5  {% endperson %}
```



## Remember:

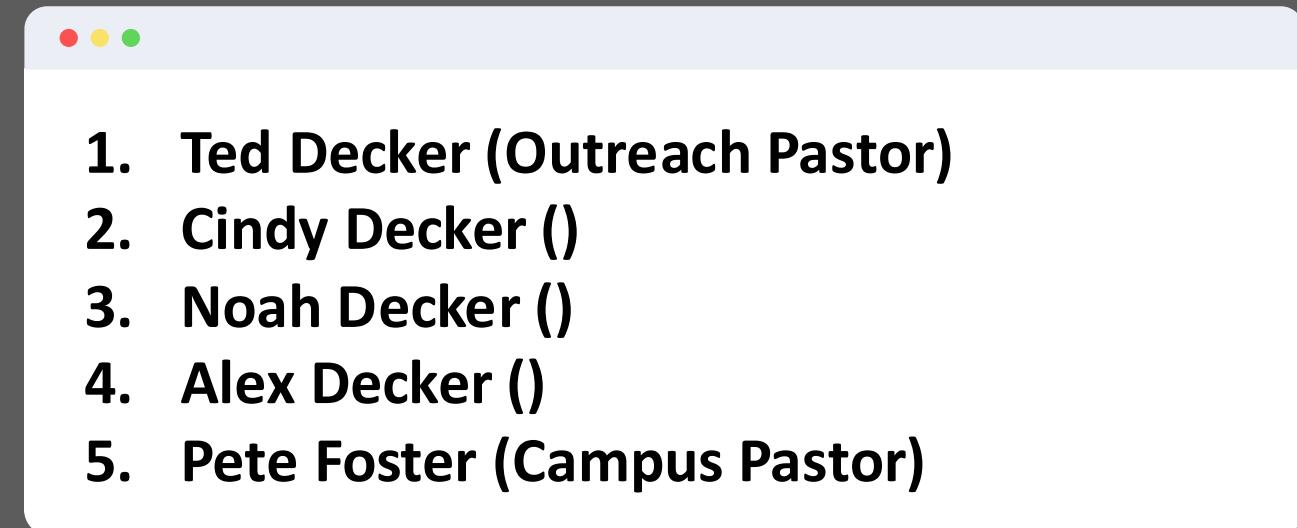
Properties available to Lava can  
be found on the Model Map





# Entity Command

```
1  {% person where:'LastName == "Decker" || Position *= "Pastor"' %}  
2    <ol>  
3      {% for person in personItems %}  
4          <li>{{ person.FullName }} ({{ person | Attribute:'Position' }})</li>  
5      {% endfor %}  
6    </ol>  
7  {% endperson %}
```





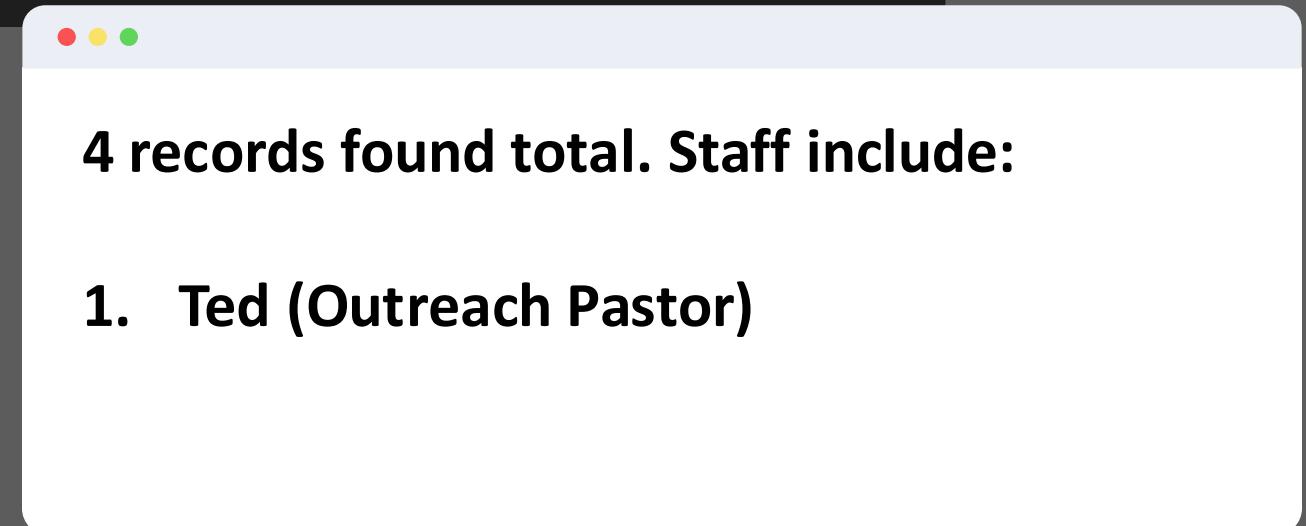
# Entity Command Operators

<code>==</code>	Is Equal To	<code>&gt;</code>	Is Greater Than
<code>!=</code>	Is Not Equal To	<code>&gt;=</code>	Is Greater Than or Equal To
<code>^=</code>	Starts With	<code>&lt;</code>	Is Less Than
<code>*=</code>	Contains	<code>&lt;=</code>	Is Less Than or Equal To
<code>*!</code>	Does Not Contain	<code>_=</code>	Is Blank
<code>\$=</code>	Ends With	<code>_!</code>	Is Not Blank
<code>&amp;&amp;</code>	Logical AND operator		
<code>  </code>	Logical OR operator		



# Entity Command

```
1  {% person where:'LastName == "Decker"' %}  
2      <p>{{ personItems | Size }} records found total. Staff include:</p>  
3      <ol>  
4          {% for person in personItems %}  
5              {% assign positionLength = person | Attribute:'Position' | Size %}  
6              {% if positionLength > 0 %}  
7                  <li>{{ person.FullName }} ({{ position }})</li>  
8              {% endif %}  
9          {% endfor %}  
10     </ol>  
11  {% endperson %}
```



The screenshot shows a standard Mac OS X window frame with red, yellow, and green control buttons at the top center. Inside the window, the text "4 records found total. Staff include:" is displayed in a large, bold, black font. Below this, the first item from the list is shown: "1. Ted (Outreach Pastor)". The background of the window is white, and the overall appearance is that of a terminal or code editor interface.

**4 records found total. Staff include:**

**1. Ted (Outreach Pastor)**



# Entity Command

Property	Attribute
1  {% person where:'LastName == "Decker" && Position != "1"' %}	
2    <p>{{ personItems   Size }} records found total. Staff Include:</p>	
3    <ol>	
4      {% for person in personItems %}	
5        <li>{{ person.FullName }} ({{ person   Attribute:'Position' }})</li>	
6      {% endfor %}	
7    </ol>	
8  {% endperson %}	

The screenshot shows a standard Mac OS X window frame with red, yellow, and green control buttons at the top. Inside the window, the text is displayed in a large, bold, black font. It starts with a summary of records found, followed by a numbered list of staff members with their names and titles.

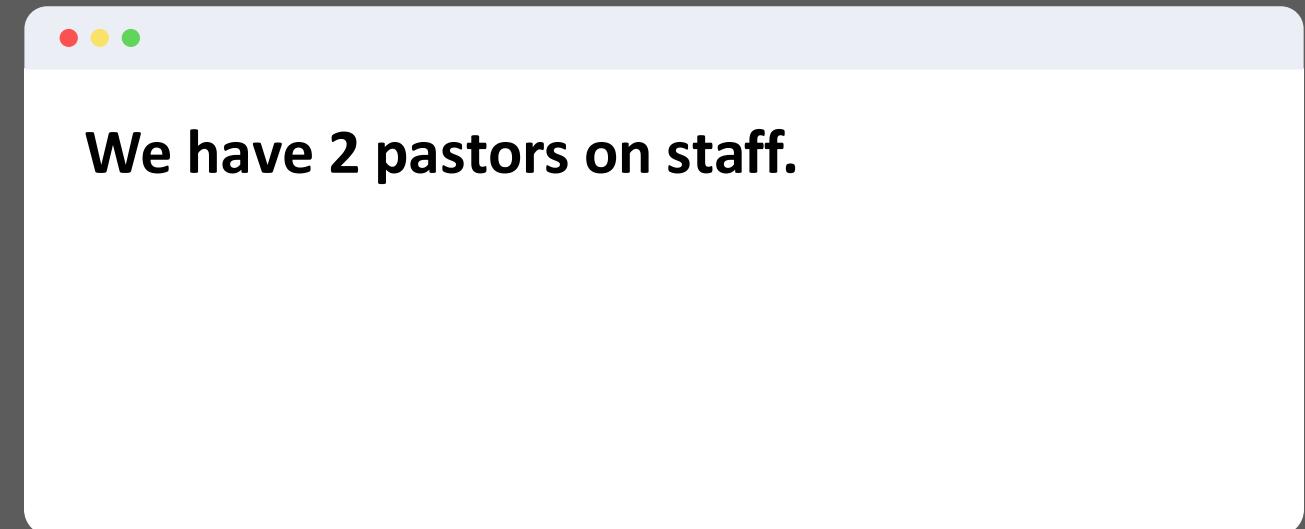
**1 records found total. Staff include:**

**1. Ted (Outreach Pastor)**



# Count

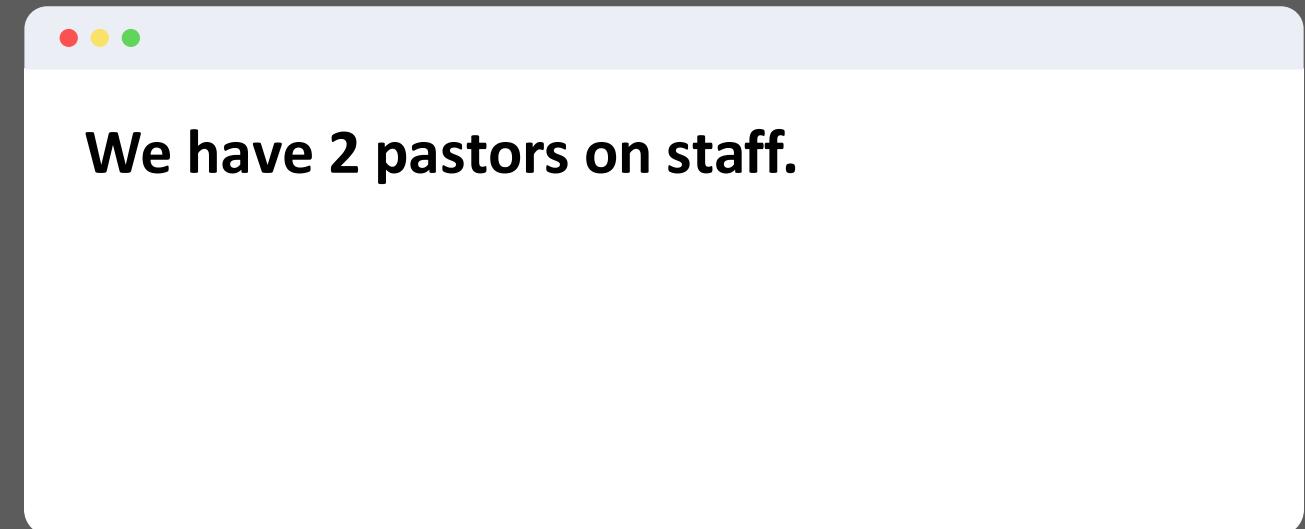
```
1  {% person where:'Position _! "1"' count:'true' %}  
2  We have {{ count }} pastors on staff.  
3  {% endperson %}
```





# Count

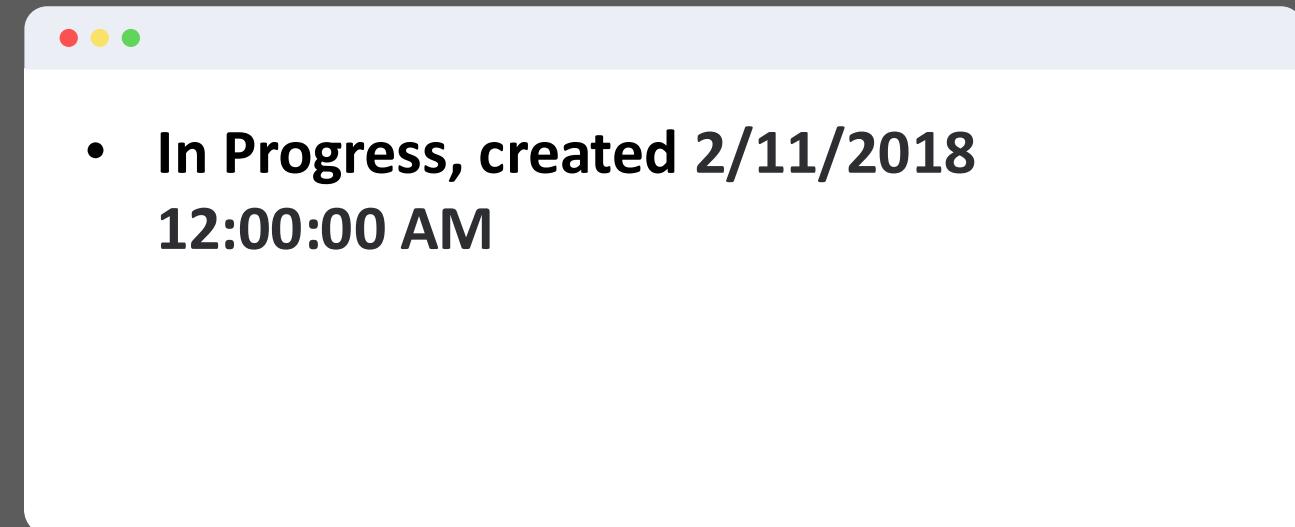
```
1  {% person where:'Position _! "1"' count:'true' %}  
2  We have {{ 'pastor' | ToQuantity:count }} on staff.  
3  {% endperson %}
```





# Entities

```
1  {% connectionrequest where:'ConnectionOpportunityId == "3" && ConnectionStateId == "0"' %}  
2    <ul>  
3      {% for request in connectionrequestItems %}  
4        <li>{{ request.ConnectionStatus.Name }}, created {{ request.CreatedDateTime }}</li>  
5      {% endfor %}  
6    </ul>  
7  {% endconnectionrequest %}
```





# Model Map

Assemble your Lava templates

The screenshot shows the Lava Model Map interface. On the left is a vertical navigation bar with icons for Home, Power Tools, and various system components like Check-in, CMS, Communication, Core, CRM, Engagement, Event, Finance, Group, LMS, Meta, Prayer, Reporting, Web Farm, and Workflow. The main area is titled "Model Map" and displays a 3x5 grid of these categories. The "Engagement" category is highlighted with a red arrow pointing to its icon.

Model Map				
Check -in	CMS	Communication	Core	CRM
Engagement	Event	Finance	Group	LMS
Meta	Prayer	Reporting	Web Farm	Workflow



# Model Map

Assemble your Lava templates

The screenshot shows the 'Model Map' interface. On the left is a vertical sidebar with icons for Home, Power Tools, Model Map, and a user profile. The main area is titled 'Model Map' and contains a grid of 15 icons representing different models: Check-in, CMS, Communication, Core, CRM, Engagement, Event, Finance, Group, LMS, Meta, Prayer, Reporting, Web Farm, and Workflow. A red arrow points from the bottom of the sidebar towards the right-hand details panel.

The screenshot shows the 'AchievementAttempt' model details page. At the top is an orange header bar with a logo, search, and user profile. Below is a navigation bar with icons for Meta, Prayer, Reporting, Web Farm, and Workflow. The main content is divided into two sections: 'Engagement Models' (left) and 'Model Details' (right). The 'Engagement Models' section lists various entities: Achievement Attempt, Achievement Type, Achievement Type Prerequisite, Connection Activity Type, Connection Opportunity, Connection Opportunity Campus, Connection Opportunity Connector, Group, Connection Opportunity Group, Connection Opportunity Group Config, Connection Request, Connection Request Activity, Connection Request Workflow, Connection Status, Connection Status Automation, Connection Type, Connection Workflow, Step, Step Program, and Step Program Completion. The 'Model Details' section is for 'AchievementAttempt'. It includes a 'Properties' table with the following columns:

Property	Description
<code>AchievementAttemptEndDateTime</code>	Gets or sets the achievement attempt end date time.
<code>AchievementAttemptStartTime</code>	Gets or sets the achievement attempt start date time.
<code>AchievementType</code>	Gets or sets the <code>AchievementType</code> of this attempt.
<code>AchievementTypeId</code>	Gets or sets the Id of the <code>AchievementType</code> to which this attempt belongs. This property is required.
<code>AchieverEntityId</code>	Gets or sets the achiever entity identifier. The



# Model Map

Assemble your Lava templates

{{ item

The screenshot shows the 'Engagement Models' section of the Model Map interface. On the left, there's a sidebar with icons for Meta, Prayer, Reporting, Web Farm, and Workflow. The main area has tabs for 'Model Details' and 'Properties'. Under 'Model Details', it shows 'AchievementAttempt' with a description: 'Represents an Achievement Attempts in Rock.' It includes 'Filter Options' and 'Show: Methods'. Under 'Properties', it lists: 'AchievementAttemptEndDateTime' (Gets or sets the achievement attempt end date time), 'AchievementAttemptStartTime' (Gets or sets the achievement attempt start date time), 'AchievementType' (Gets or sets the AchievementType of this attempt), 'AchievementTypeId' (Gets or sets the Id of the AchievementType to which this attempt belongs. This property is required.), and 'AchieverEntityId' (Gets or sets the achiever entity identifier. The).

The screenshot shows two side-by-side views of the Model Map interface. The left view shows the 'Engagement Models' section with a list of entities: Achievement Attempt, Achievement Type, Achievement Type Prerequisite, Connection Activity Type, Connection Opportunity, Connection Opportunity Campus, Connection Opportunity Connector Group, Connection Opportunity Group, Connection Opportunity Group Config, Connection Request, Connection Request Activity, Connection Request Workflow, Connection Status, Connection Status Automation, Connection Type, Connection Workflow, Step, Step Program, Step Program Completion, and Step Status. The right view shows the 'Model Details' section for 'ConnectionRequest' with a description: 'Represents a connection request'. It includes 'Filter Options' and 'Show: Methods'. Under 'Properties', it lists: 'AdditionalLavaFields', 'AssignedGroup' (Gets or sets the assigned Group), 'AssignedGroupId' (Gets or sets the assigned Group identifier), 'AssignedGroupMemberAttributeValues' (Gets or sets the assigned group member attribute values), 'AssignedGroupMemberRoleId' (Gets or sets the assigned group member role identifier), 'AssignedGroupMemberStatus' (Gets or sets the assigned group member status), and 'Attributes'.



# Model Map

Assemble your Lava templates

`{{ item.ConnectionStatus}}`

The screenshot shows the Lava Model Map interface. On the left is a navigation sidebar with icons for Home, Meta, Prayer, Reporting, Web Farm, and Workflow. The main area displays the 'Engagement Models' section under 'Achievement Attempt'. It includes a 'Model Details' panel with a 'Filter Options' dropdown and a 'Show Methods' checkbox. Below this is a detailed description of the 'AchievementAttempt' class, which represents an Achievement Attempts in Rock. It lists properties: 'AchievementAttemptEndDateTime' (Gets or sets the achievement attempt end date time), 'AchievementAttemptStartTime' (Gets or sets the achievement attempt start date time), 'AchievementType' (Gets or sets the AchievementType of this attempt), 'AchievementTypeId' (Gets or sets the Id of the AchievementType to which this attempt belongs. This property is required), and 'AchieverEntityId' (Gets or sets the achiever entity identifier. The).

The screenshot shows the Lava Model Map interface. On the right is a list of properties for the 'ConnectionState' class. A red arrow points from the 'AchievementAttempt' model details in the previous screenshot to the 'ConnectionStatus' property. Another red arrow points from the 'AchievementAttempt' model details to the 'CreatedDateTime' property.

<code> ConnectionState *</code>	Gets or sets the state of the connection. This is a hard coded list of values defined in the code as an enumeration.
<code> ConnectionStatus</code>	Gets or sets the <code>ConnectionStatus</code> .
<code> ConnectionStatusId *</code>	Gets or sets the <code>ConnectionStatus</code> identifier.
<code> ConnectionTypeId *</code>	Gets or sets the <code>ConnectionType</code> identifier.
<code> ConnectorPersonAlias</code>	Gets or sets the connector <code>PersonAlias</code> .
<code> ConnectorPersonAliasId</code>	Gets or sets the connector <code>PersonAlias</code> identifier.
<code> ContextKey</code>	
<code> CreatedByPersonAlias</code>	
<code> CreatedByPersonAliasId</code>	
<code> CreatedByPersonId</code>	
<code> CreatedByPersonName</code>	
<code> CreatedDateKey</code>	Gets the created date key.
<code> CreatedDateTime</code>	



# Model Map

Assemble your Lava templates

The screenshot shows the Model Map application interface. On the left, there's a sidebar with various icons and a main content area titled "Engagement Models". A red arrow points from the sidebar towards the "Engagement Models" list. The main content area has tabs for "Meta", "Prayer", "Reporting", "Web Farm", and "Workflow". The "Reporting" tab is active. Below the tabs, the "Engagement Models" list includes: Achievement Attempt, Achievement Type, Achievement Type Prerequisite, Connection Activity Type, Connection Opportunity, Connection Opportunity Campus, Connection Opportunity Connector Group, Connection Opportunity Group, Connection Opportunity Group Config, Connection Request, Connection Request Activity, Connection Request Workflow, Connection Status, Connection Status Automation, Connection Type, Connection Workflow, Step, Step Program, and Step Program Completion. To the right, the "Model Details" section is displayed for "AchievementAttempt". It shows a description: "Represents an Achievement Attempts in Rock.", a "Properties" table, and five listed properties: AchievementAttemptEndDateTime, AchievementAttemptStartTime, AchievementType, AchievementTypeId, and AchieverEntityId.

Property	Description
AchievementAttemptEndDateTime	Gets or sets the achievement attempt end date time.
AchievementAttemptStartTime	Gets or sets the achievement attempt start date time.
AchievementType	Gets or sets the AchievementType of this attempt.
AchievementTypeId	Gets or sets the Id of the AchievementType to which this attempt belongs. This property is required.
AchieverEntityId	Gets or sets the achiever entity identifier. The



# Model Map

Assemble your Lava templates

`{{ item.ConnectionStatus }}`

The screenshot shows the 'Engagement Models' section of the application. On the left is a sidebar with icons for Meta, Prayer, Reporting, Web Farm, and Workflow. The main area displays the 'Model Details' for 'AchievementAttempt'. It includes a 'Properties' table with columns for name, description, and type. Properties listed include 'AchievementAttemptEndDateTime', 'AchievementAttemptStartTime', 'AchievementType', 'AchievementTypeId', and 'AchieverEntityId'.

The screenshot shows the 'Model Details' section for 'ConnectionStatus'. It includes a 'Properties' table with columns for name, description, and type. Properties listed include 'AdditionalLavaFields', 'Attributes', 'AttributeValueDefaults', 'AttributeValues', 'AutoInactivateState', 'AvailableKeys', and 'ConnectionStatusAutomations'.



# Model Map

Assemble your Lava templates

`{{ item.ConnectionStatus.Name }}`

The screenshot shows the Engagement Models section of the application. On the left is a sidebar with icons for Meta, Prayer, Reporting, Web Farm, and Workflow. The main area displays the 'Engagement Models' list, which includes: Achievement Attempt, Achievement Type, Achievement Type Prerequisite, Connection Activity Type, Connection Opportunity, Connection Opportunity Campus, Connection Opportunity Connector Group, Connection Opportunity Group, Connection Opportunity Group Config, Connection Request, Connection Request Activity, Connection Request Workflow, Connection Status, Connection Status Automation, Connection Type, Connection Workflow, Step, Step Program, and Step Program Completion. On the right, the 'Model Details' pane is open for 'AchievementAttempt'. It shows the class name 'AchievementAttempt' and its properties: 'AchievementAttemptEndDateTime', 'AchievementAttemptStartTime', 'AchievementType', 'AchievementTypeId', and 'AchieverEntityId'. Each property has a brief description and a small orange lightning bolt icon indicating it's a required field.

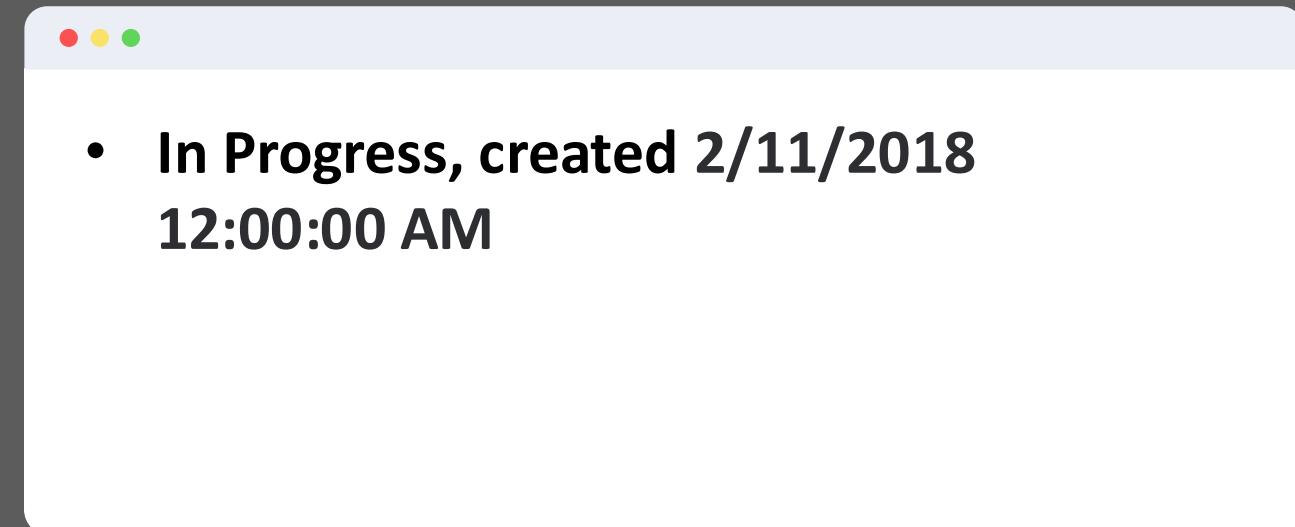
The screenshot shows the Model Map interface. On the left is a large black vertical bar. To its right is a list of properties for the 'AchievementAttempt' class. A red arrow points from the 'Name' property in the list to the 'Name' property in the 'Model Details' pane of the previous screenshot. The properties listed are: IsValid, Item, ModifiedAuditValuesAlreadyUpdated, ModifiedByPersonAlias, ModifiedByPersonAliasId, ModifiedByPersonId, ModifiedByPersonName, ModifiedDateTime, Name (with a red asterisk and lightning bolt icon), Order, ParentAuthority, ParentAuthorityPre, SupportedActions, TypeId, and TypeName. Each property is accompanied by a brief description and a small orange lightning bolt icon.

Property	Description
IsValid	
Item	
ModifiedAuditValuesAlreadyUpdated	
ModifiedByPersonAlias	
ModifiedByPersonAliasId	
ModifiedByPersonId	
ModifiedByPersonName	
ModifiedDateTime	
Name *	Gets or sets the name.
Order	Gets or sets the order.
ParentAuthority	
ParentAuthorityPre	
SupportedActions	
TypeId	
TypeName	



# Entities

```
1  {% connectionrequest where:'ConnectionOpportunityId == "3" && ConnectionStateId == "0"' %}  
2    <ul>  
3      {% for request in connectionrequestItems %}  
4        <li>{{ request.ConnectionStatus.Name }}, created {{ request.CreatedDateTime }}</li>  
5      {% endfor %}  
6    </ul>  
7  {% endconnectionrequest %}
```





# Entities

```
1  {% group where:'GroupTypeId == "10"' %}  
2    <ul>  
3      {% for g in groupItems %}  
4          <li>{{ g.Name }}</li>  
5      {% endfor %}  
6    </ul>  
7  {% endgroup %}
```

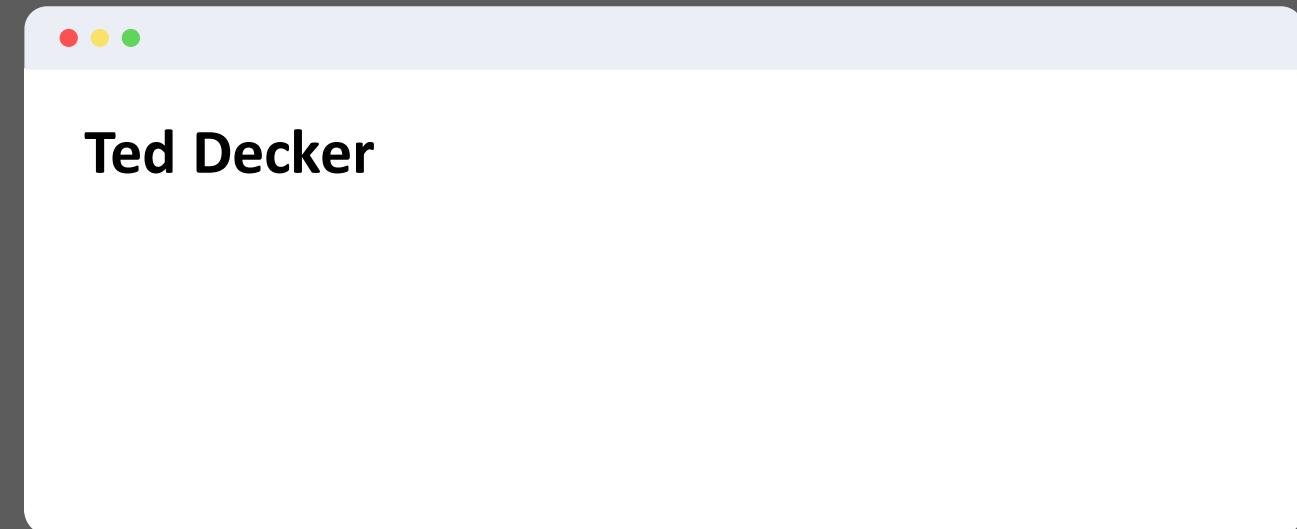
The screenshot shows a Mac OS X style application window with a white background and a title bar at the top. The title bar has three colored circular buttons (red, yellow, green) on the right side. The main content area contains a bulleted list of five family names, each displayed in a bold black font. The list items are:

- **Decker Family**
- **Jones Family**
- **Simmons Family**
- **Jackson Family**
- **Lowe Family**



# Id

```
1  {% person id:'5' %}  
2      {% for person in personItems %}  
3          {{ person.FullName }}  
4      {% endfor %}  
5  {% endperson %}
```





Id

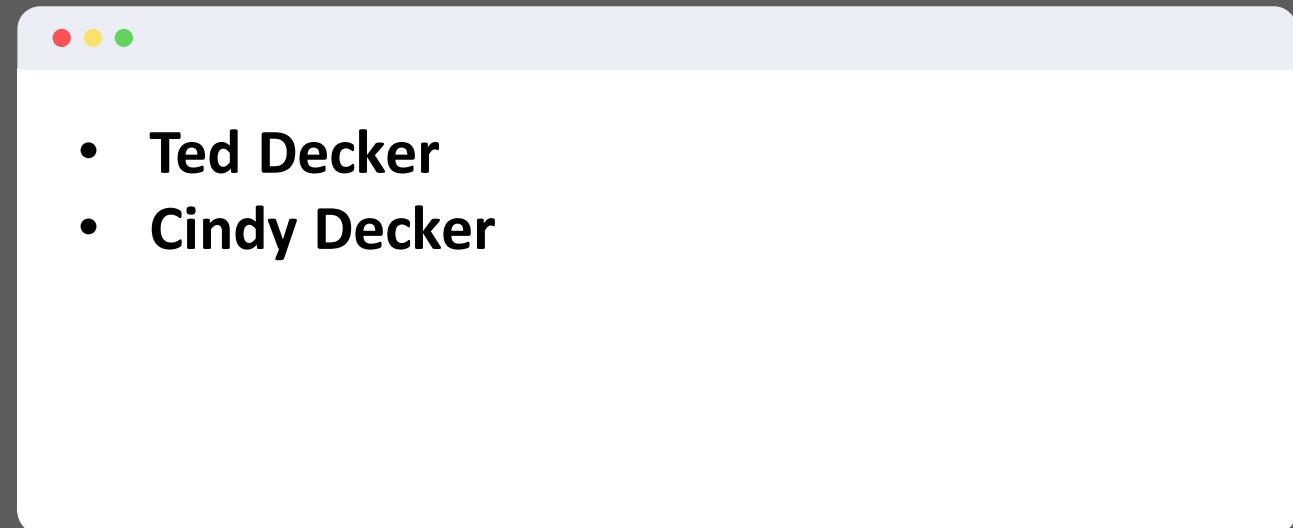
```
1  {% person id:'5' %}  
2    {{ person.FullName }}  
3  {% endperson %}
```





# Id

```
1  {% person ids:'5,6' %}  
2    <ul>  
3      {% for p in personItems %}  
4          <li>{{ p.FullName }}</li>  
5      {% endfor %}  
6    </ul>  
7  {% endperson %}
```





# Dataview

```
1  {% person dataview:'1' %}  
2    <ul>  
3      {% for p in personItems %}  
4          <li>{{ p.FullName }}</li>  
5      {% endfor %}  
6    </ul>  
7  {% endperson %}
```





# Dataview

```
1  {% person dataview:'1' where:'Gender == "Female"' %}  
2    <ul>  
3      {% for p in personItems %}  
4        <li>{{ p.FullName }}</li>  
5      {% endfor %}  
6    </ul>  
7  {% endperson %}
```

The screenshot shows a Mac OS X application window with a white background and a title bar featuring the standard red, yellow, and green buttons. Inside the window, there is a single bullet-pointed list of five names, all displayed in a large, bold, black font:

- Cindy Decker
- Sarah Simmons
- Mariah Jackson
- Alisha Marble
- Pamela Foster



# Sort

```
1  {% person dataview:'1' where:'Gender == "Female"' sort:'LastName, FirstName desc' %}  
2    <ul>  
3      {% for p in personItems %}  
4        <li>{{ p.FullName }}</li>  
5      {% endfor %}  
6    </ul>  
7  {% endperson %}
```





# Offset and Limit

```
1  {% person dataview:'1' where:'Gender == "Female"' sort:'LastName, FirstName desc' offset:'5' limit:'5' %}  
2    <ul>  
3      {% for p in personItems %}  
4        <li>{{ p.FullName }}</li>  
5      {% endfor %}  
6    </ul>  
7  {% endperson %}
```





# Offset and Limit

```
1  {% person
2      dataview:'1'
3      where:'Gender == "Female"'
4      sort:'LastName, FirstName desc'
5      offset:'5'
6      limit:'5' %}
7  <ul>
8      {% for p in personItems %}
9          <li>{{ p.FullName }}</li>
10     {% endfor %}
11 </ul>
12 {% endperson %}
```



The screenshot shows a window with a title bar containing three colored dots (red, yellow, green). The main content area displays a bulleted list of five names:

- **Jenny Michaels**
- **Becky Peterson**
- **Sarah Simmons**
- **Corrie Stone**
- **Nancy Sweeny**



# Security

```
1  {% group where:'GroupTypeId == "25" %}  
2    <ul>  
3      {% for g in groupItems %}  
4          <li>{{ g.Name }}</li>  
5      {% endfor %}  
6    </ul>  
7  {% endperson %}
```





# Security

```
1  {% group where:'GroupTypeId == "25" securityenabled:'false' %}  
2    <ul>  
3      {% for g in groupItems %}  
4          <li>{{ g.Name }}</li>  
5      {% endfor %}  
6    </ul>  
7  {% endperson %}
```





# Iterator

```
1  {% group where:'GroupTypeId == "25" securityenabled:'false' iterator:'Groups' %}
2  <ul>
3      {% for group in Groups %}
4          <li>{{ group.Name }}</li>
5      {% endfor %}
6  </ul>
7  {% endperson %}
```





# Power Tools

For Speed and Cross-Entity Filtering





# Expression

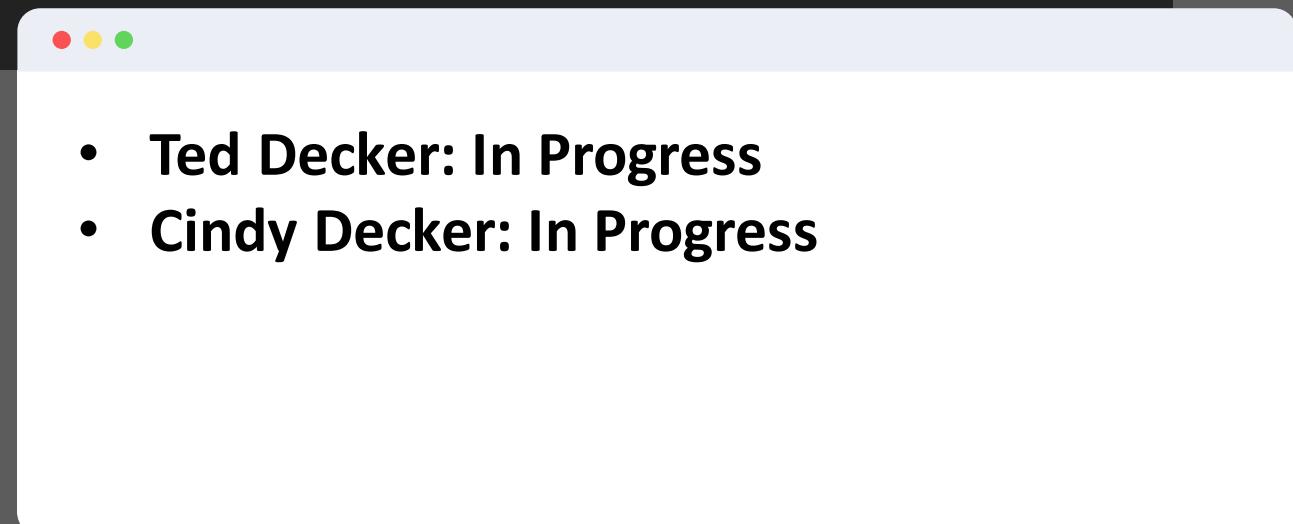
```
1  {% step where:'StepTypeId == 2'
2      expression:'PersonAlias.Person.AgeClassification == 1'
3      securityenabled:'false' %}

4

5  {% for step in stepItems %}
6      <ul>
7          <li>{{ step.PersonAlias.Person.FullName }}: {{ step.StepStatus }}</li>
8      </ul>
9  {% endfor %}

10

11  {% endstep %}
```





# Expression

```
1  {% step where:'StepTypeId == 2'  
2      expression:'PersonAlias.Person.AgeClassification == 1'  
3      securityenabled:'false' %}  
4  
5      {% for step in stepItems %}  
6          <ul>  
7              <li>{{ step.PersonAlias.Person.FullName }}: {{ step.StepStatus }}</li>  
8          </ul>  
9      {% endfor %}  
10  
11  {% endstep %}
```





# Lazy Loading

```
1  {% step where:'StepTypeId == 2'  
2      expression:'PersonAlias.Person.AgeClassification == 1'  
3      securityenabled:'false'  
4      lazyloadenabled:'false' %}  
5  
6  {% for step in stepItems %}  
7      <ul>  
8          <li>{{ step.Id }} {{ step.PersonAlias.Person.FullName }}: {{ step.StepStatus }}</li>  
9      </ul>  
10     {% endfor %}  
11  
12  {% endstep %}
```





# Include

```
1  {% step where:'StepTypeId == 2'  
2      expression:'PersonAlias.Person.AgeClassification == 1'  
3      securityenabled:'false'  
4      lazyloadenabled:'false'  
5      include:'PersonAlias.Person, StepStatus' %}  
6  
7  {% for step in stepItems %}  
8      <ul>  
9          <li>{{ step.PersonAlias.Person.FullName }}: {{ step.StepStatus }}</li>  
10     </ul>  
11  {% endfor %}  
12  
13  {% endstep %}
```





# Select

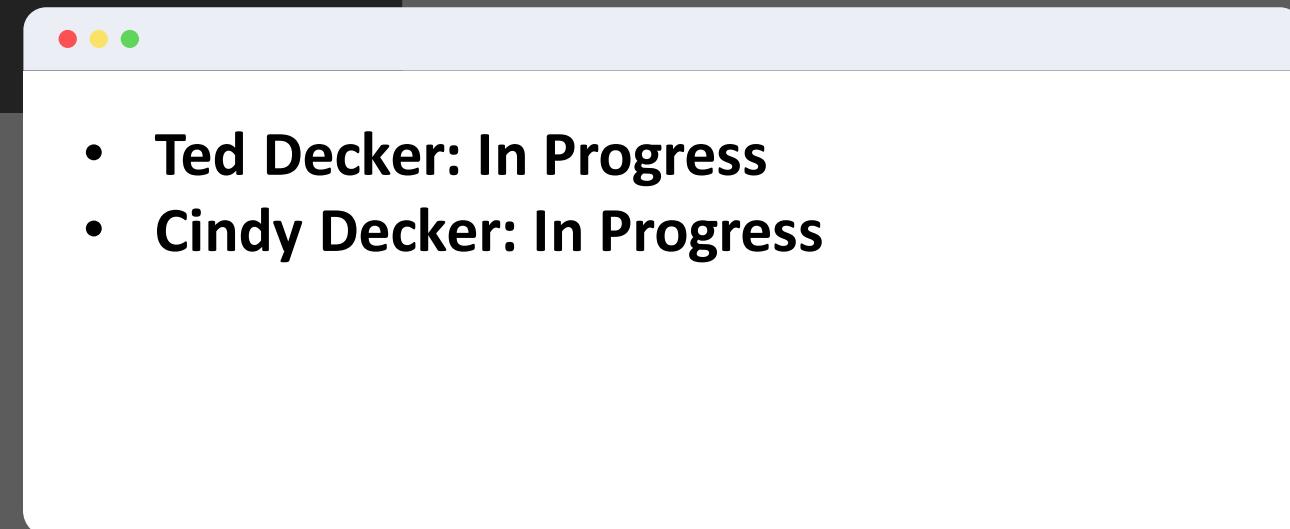
```
1  {% step where:'StepTypeId == 2'  
2      expression:'PersonAlias.Person.AgeClassification == 1'  
3      select:'new( PersonAlias.Person.NickName as NickName,  
4                  PersonAlias.Person.LastName as LastName,  
5                  StepStatus.Name as Status)'  
6      securityenabled:'false' %}  
7  
8  {% for step in stepItems %}  
9      <ul>  
10         <li>{{ step.NickName }} {{ step.LastName }}: {{ step.Status }}</li>  
11     </ul>  
12  {% endfor %}  
13  
14  {% endstep %}
```





# Select

```
1  {% step where:'StepTypeId == 2'
2      expression:'PersonAlias.Person.AgeClassification == 1'
3      select:'new{ PersonAlias.Person.FullName as PersonName,
4                  StepStatus as Status}'
5      securityenabled:'true' %}
6
7  {% for step in stepItems %}
8      <ul>
9          <li>{{ step.PersonName }}: {{ step.Status }}</li>
10     </ul>
11  {% endfor %}
12
13 {% endstep %}
```





# Select

```
1  {% step where:'StepTypeId == 2'
2      expression:'PersonAlias.Person.AgeClassification == 1'
3      select:'new( PersonAlias.Person.NickName as NickName,
4                  PersonAlias.Person.LastName as LastName,
5                  StepStatus.Name as Status)'
6      securityenabled:'false' %}
7
8  <pre>{{ stepItems | ToJSON }}</pre>
9
10 {% endstep %}
```

```
1  [
2    {
3      "NickName": "Ted",
4      "LastName": "Decker",
5      "Status": "In Progress"
6    },
7    {
8      "NickName": "Cindy",
9      "LastName": "Decker",
10     "Status": "In Progress"
11   }
12 ]
```



# Select

```
1  {% step where:'StepTypeId == 2'
2      expression:'PersonAlias.Person.AgeClassification == 1'
3      select:'new{ PersonAlias.Person.NickName as NickName,
4                  PersonAlias.Person.LastName as LastName,
5                  PersonAlias.Person.PhoneNumbers.Where( NumberTypeValueId == 12 ).Select( new( NumberFormatted as Number ) ) as Phones,
6                  StepStatus.Name as Status}'
7      securityenabled:'false' %}
8
9  {% for step in stepItems %}
10     <ul>
11         <li>{{ step.NickName }} {{ step.LastName}}: {{ step.Phones | First | Property:'Number' }}</li>
12     </ul>
13
14    {% endfor %}
15
16  {% endstep %}
```

The screenshot shows a window with a title bar and three red, yellow, and green buttons. The main content area contains a bulleted list of names and phone numbers:

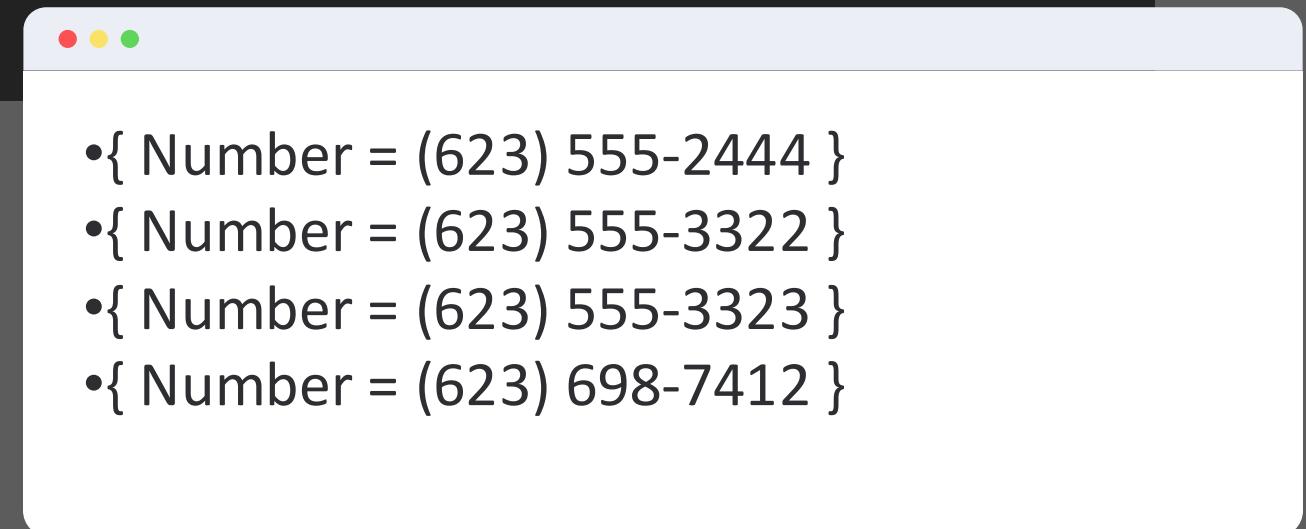
- **Ted Decker: (623) 555-3322**
- **Cindy Decker: (623) 555-3323**



# Select Many

```
1  {% step where:'StepTypeId == 2'
2      expression:'PersonAlias.Person.AgeClassification == 1'
3      selectmany:'PersonAlias.Person.PhoneNumbers.Select( new( NumberFormatted as Number )).Distinct()'
4      securityenabled:'false' %}

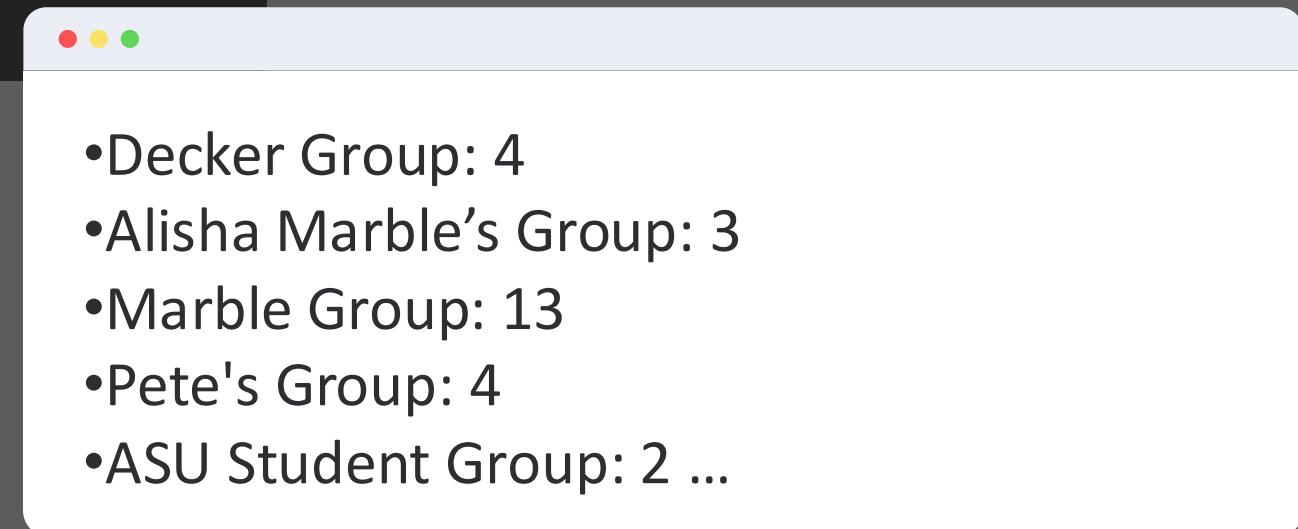
5
6  <ul>
7      {% for step in stepItems %}
8          <li>
9              {{ step }}
10         </li>
11     {% endfor %}
12   </ul>
13
14  {% endstep %}
```





# Aggregates

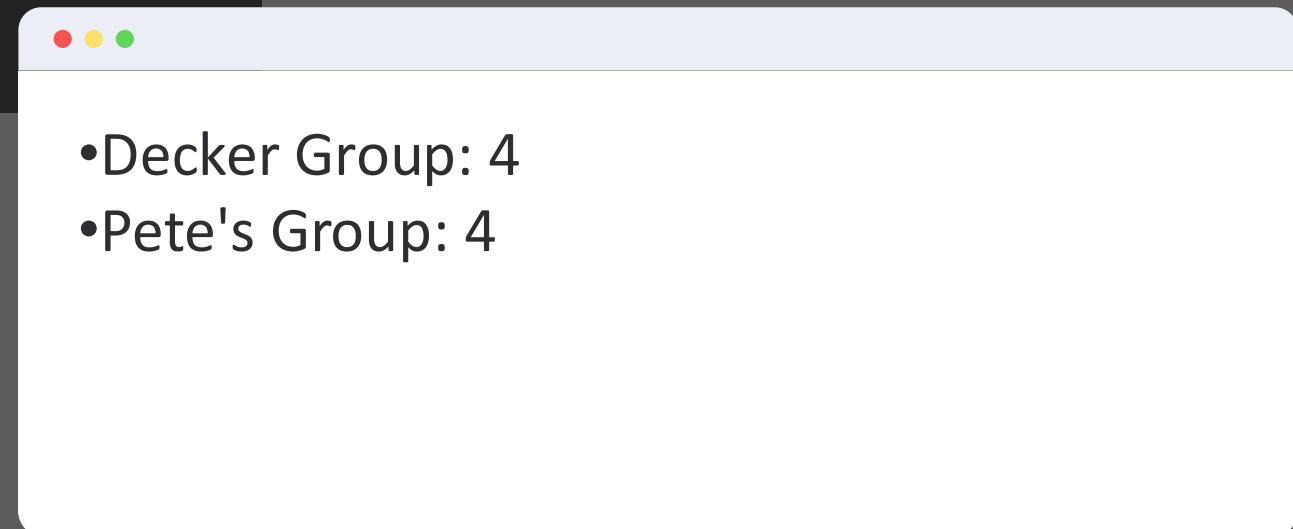
```
1  {% group where:'GroupTypeId == 25'  
2      select:'new( Name, Members.Count() as MemberCount )'  
3      securityenabled:'false' %}  
  
4  
5  {% for group in groupItems %}  
6      <ul>  
7          <li>  
8              {{ group.Name }}: {{ group.MemberCount }}  
9          </li>  
10         </ul>  
11     {% endfor %}  
12  
13  {% endgroup %}
```





# Aggregates

```
1  {% group where:'GroupTypeId == 25'  
2      expression:'Members.Count() == 4'  
3      select:'new( Name, Members.Count() as MemberCount )'  
4      securityenabled:'false' %}  
5  
6  {% for group in groupItems %}  
7      <ul>  
8          <li>  
9              {{ group.Name }}: {{ group.MemberCount }}  
10         </li>  
11     </ul>  
12  {% endfor %}  
13  
14  {% endgroup %}
```





# Groupby

```
1  {% groupmember groupby:'new( GroupId, Group.Name)'  
2      where:'GroupTypeId == 10'  
3      select:'new( Key as Key, It as GroupMembers)'  
4      securityenabled:'false'  
5      limit:3 %}  
6  
7      {% for family in groupmemberItems %}  
8          {{ family.Key.Name }}  
9          <ul>  
10             {% for member in family.GroupMembers %}  
11                 <li>{{ member.Person.NickName }} {{ member.Person.LastName }}</li>  
12             {% endfor %}  
13         </ul>  
14     {% endfor %}  
15  
16  
17  {% endgroupmember %}
```

The screenshot shows a Mac OS X application window with three colored window controls (red, yellow, green) at the top right. The main content area displays a list of user names:

- Admin
- Admin Admin
- Anonymous Family
- Giver Anonymous
- Presence Family
- Presence Presence



# Groupby

```
1  {% groupmember groupby:'new( GroupId, Group.Name )'
2      where:'GroupTypeId == 10'
3      select:'new( Key as Key,
4                  Select( new(Person.Nickname as NickName, Person.LastName as LastName ) ) as GroupMembers )'
5      securityenabled:'false'
6      limit:3 %}
7
8  {% for family in groupmemberItems %}
9      <p>{{ family.Key.Name }}</p>
10     <ul>
11         {% for member in family.GroupMembers %}
12             <li>{{ member.NickName }} {{ member.LastName }}</li>
13         {% endfor %}
14     </ul>
15 {% endfor %}
16
17 {% endgroupmember %}
```

A screenshot of a Mac OS X application window titled "Admin". The window contains a list of family members under the heading "Anonymous Family".

Admin

- Admin Admin
- Anonymous Family
- Giver Anonymous
- Presence Family
- Presence Presence



# Groupby with Aggregates

```
1  {% groupmember groupby:'new(GroupId, Group.Name)'  
2          where:'GroupTypeId == 10'  
3          select:'new( Key as Key, Count() as MemberCount )'  
4          securityenabled:'false'  
5          limit:3 %}  
6  
7  {% for family in groupmemberItems %}  
8      {{ family.Key.Name }}: {{ family.MemberCount }}<br/>  
9  
10 {% endfor %}  
11  
12 {% endgroupmember %}
```

Admin: 1  
Anonymous Family: 1  
Presence Family: 1



# My attempts to show the expression clause are getting too complicated

I think I need to throw away everything below and go back to the approach  
Jon used in <https://www.triumph.tech/videos/entity-commands-20>

Maybe get all of a person's connection request activities, to give variety  
over the FinancialTransaction examples that have been done...?



# “How can I...”

... get a person's Financial Transactions?

The screenshot shows the Model Map interface within a larger application. On the left is a vertical navigation bar with icons for Home, Power Tools, Model Map, CMS, Communication, Core, CRM, Engagement, Event, Finance, Group, Meta, Prayer, Reporting, Web Farm, and Workflow. The 'Power Tools' icon is highlighted with a red border. The main area is titled 'Model Map' and shows a grid of ten boxes, each representing a different system component. A red arrow points to the 'Finance' box, which contains a money bag icon.

Model Map

Home > Power Tools > Model Map

Model Map

</> CMS	Communication	Core	CRM	Engagement
Event	Finance	Group	Meta	Prayer
Reporting	Web Farm	Workflow		



# “How can I...”

## ... get a person’s Financial Transactions?

The screenshot shows a software application interface with an orange header bar containing a logo, a search bar, and user profile information. Below the header is a navigation bar with several icons: Event, Finance (selected), Group, Meta, Prayer, Reporting, Web Farm, and Workflow. On the far left is a vertical sidebar with icons for Home, People, Finance, Tools, and Reports. The main content area is divided into two sections: 'Finance Models' on the left and 'Model Details' on the right.

**Finance Models**

- Benevolence Request
- Benevolence Request Document
- Benevolence Result
- Benevolence Type
- Benevolence Workflow
- Financial Account
- Financial Batch
- Financial Gateway
- Financial Payment Detail
- Financial Person Bank Account
- Financial Person Saved Account
- Financial Pledge
- Financial Scheduled Transaction
- Financial Scheduled Transaction Detail
- Financial Statement Template
- Financial Transaction** (highlighted with a red arrow)
- Financial Transaction Alert
- Financial Transaction Alert Type

**Model Details**

Filter Options Show:  Methods

**FinancialTransaction**

FinancialTransaction Logic

Properties

- AdditionalLavaFields
- Attributes** ⚡ (highlighted with a red arrow)
- AttributeValueDefaults
- AttributeValues ⚡
- AuthorizedPersonAlias** ⚡ (highlighted with a red arrow)
- AuthorizedPersonAliasId ⚡
- AvailableKeys

Gets or sets the authorized PersonAlias.

Gets or sets the authorized person identifier.



# “How can I...”

... get a person's Financial Transactions?

The screenshot shows the Model Map interface within a larger application. On the left is a vertical navigation bar with icons for Home, Power Tools, Model Map, CMS, Communication, Core, CRM, Engagement, Event, Finance, Group, Meta, Prayer, Reporting, Web Farm, and Workflow. The Model Map section is currently active. The main area displays a grid of 15 boxes representing different system components:

Component	Description
CMS	
Communication	
Core	
CRM	
Engagement	
Event	
Finance	
Group	
Meta	
Prayer	
Reporting	
Web Farm	
Workflow	

A red arrow points to the "CRM" component box.



# “How can I...”

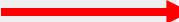
... get a person’s Financial Transactions?

The screenshot shows the Rock Software interface with a dark theme. On the left is a vertical navigation bar with icons for Home, People, Finance, Tools, and a briefcase. The main header is "Model Map" with a breadcrumb "Home > Power Tools > Model Map". The top right features a search bar, a user profile, and a dropdown menu. The central area is titled "Model Map" and contains a grid of 15 boxes representing different system modules: CMS, Communication, Core, CRM (highlighted in grey), Engagement, Event, Finance, Group, Meta, Prayer, Reporting, Web Farm, and Workflow. Below this is a "CRM Models" section listing "Assessment", "Assessment Type", "Background Check", and "Badge". To the right is a "Model Details" section for "PersonAlias", which defines it as "Represents the merge history for people in Rock. When a person is found to have a duplicate".



# “How can I...”

## ... get a person’s Financial Transactions?



■ ForeignKey	⚡
■ Guid	⚡
■ Id	⚡
■ IdKey	⚡
■ InternalMessage	⚡
Gets or sets the internal message. This is used by core Rock to track internal state messages of the record.	
■ IsValid	
■ Item	
■ LastVisitDateTime	⚡
Gets or sets the last visit time.	
■ Name	⚡
Gets or sets the name of the alias	
■ Person	⚡
Gets or sets the person.	
■ PersonId	• ⚡
Gets or sets the person Id of the Person. This property is required.	
■ TypeId	⚡
■ TypeName	⚡
■ UrlEncodedKey	⚡
■ ValidationResults	



# “How can I...”

... get a person's Financial Transactions?

The screenshot shows a software interface with a navigation bar at the top featuring three tabs: Reporting, Web Farm, and Workflow. On the left, there is a sidebar with a dark background containing a list of 'CRM Models'. On the right, the main area is titled 'Model Details' and displays information about the 'Person' model.

**CRM Models**

- Assessment
- Assessment Type
- Background Check
- Badge
- Identity Verification
- Identity Verification Code
- Person
- Person Alias
- Person Duplicate
- Person Previous Name
- Person Search Key
- Person Viewed
- Personal Device
- Phone Number
- User Login

**Key**

- A required field.
- A property on the database.
- Not mapped to the database.  
These fields are computed and are

**Model Details**

Filter Options ▾ Show:  Methods

**Person**

Represents a person or a business in Rock.

**Properties**

**AccountProtectionProfile** ⚡ Gets or sets the person's account protection profile, which is used by the duplication detection and merge processes. This is a hard coded list of values defined in the code as an enumeration.  
[Show Values ▾](#)

**AdditionalLavaFields**

**Age** ⚡ Gets the Person's age.

**AgeBracket** ⚡ Gets or sets the age bracket. This is a hard coded list of values defined in the code as an enumeration.  
[Show Values ▾](#)

**AgeClassification** ⚡ Gets or sets the age classification of the



# “How can I...”

... get a person's Financial Transactions?

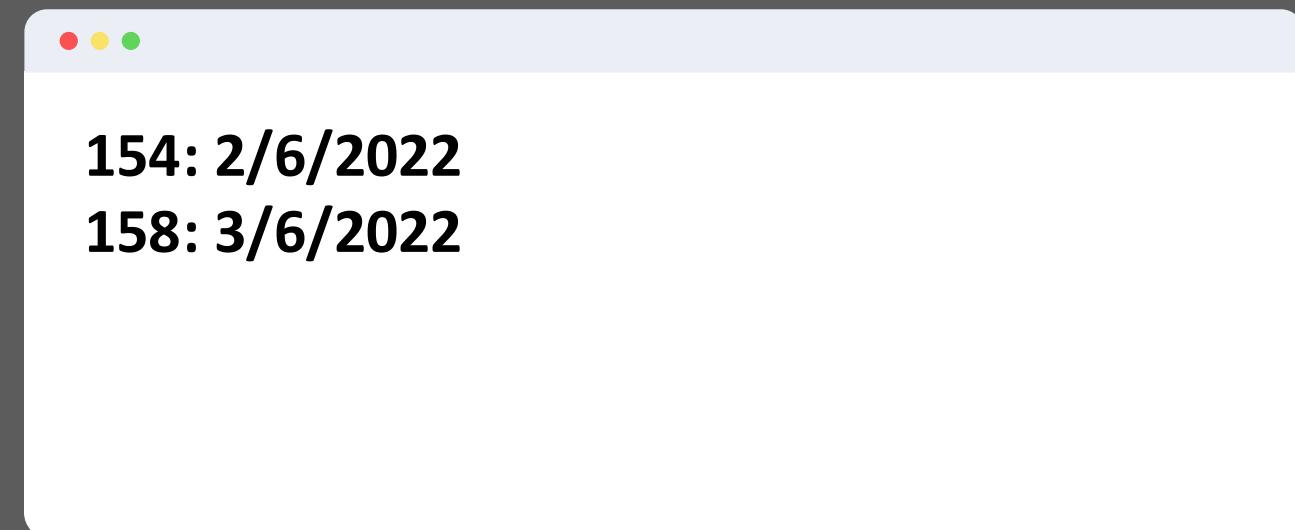


■ <b>Photo</b> ⚡	Gets or sets the <a href="#">BinaryFile</a> that contains the Person's photo.
■ <b>PhotoId</b> ⚡	Gets or sets the Id of the <a href="#">BinaryFile</a> that contains the photo of the Person.
■ <b>PhotoUrl</b> ⚡	Gets the URL of the person's photo.
■ <b>PreferredLanguageValue</b> ⚡	Gets or sets the <a href="#">DefinedValue</a> representing the Person's preferred language.
■ <b>PreferredLanguageValueId</b> ⚡	Gets or sets the <a href="#">DefinedValueId</a> of the <a href="#">DefinedValue</a> that represents the Preferred Language for this person.
■ <b>PrimaryAlias</b> ⚡	Gets the <a href="#">primary alias</a> .
■ <b>PrimaryAliasId</b> ⚡	Gets the <a href="#">primary alias</a> identifier.
■ <b>PrimaryCampus</b> ⚡	Gets or sets the person's <a href="#">primary campus</a> .
■ <b>PrimaryCampusId</b> ⚡	Gets or sets the campus id for the primary family. Note: This is computed on save, so any manual changes to this will be ignored.
■ <b>PrimaryFamily</b> ⚡	Gets or sets the <a href="#">primary family</a> .
■ <b>PrimaryFamilyId</b> ⚡	Gets or sets the group id for the <a href="#">PrimaryFamily</a> . Note: This is computed on save, so any manual changes to this will be ignored.
■ <b>PronunciationNote</b> ⚡	Gets or sets the notes for the pronunciation.



# Partial Solution

```
1  {% financialtransaction
2      where:'AuthorizedPersonAliasId == "{{ CurrentPerson.PrimaryAliasId }}"'
3      iterator:'Transactions'
4      securityenabled:'false'
5  %}
6  {% for transaction in Transactions %}
7      <p>{{ transaction.Id }}: {{ transaction.TransactionDateTime | Date:'M/d/yyyy' }}</p>
8  {% endfor %}
9  {% endfinancialtransaction %}
```





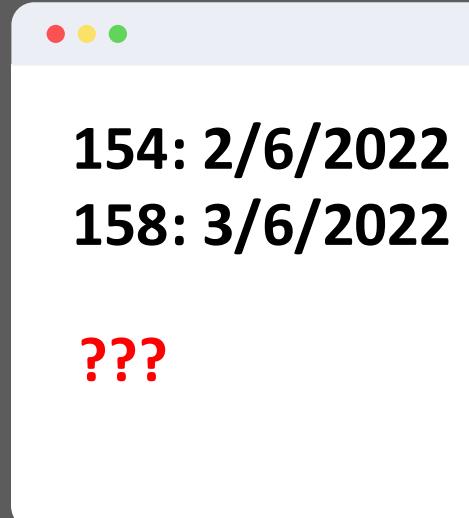
# Partial Solution

Transaction List

Enabled Filters  
Date Range: 2/1/2022 to 6/1/2022

<input type="checkbox"/>	Person	Date	Amount	Currency Type	Transaction Code	Batch Id	Accounts	Summary
<input type="checkbox"/>	Decker, Ted	2/6/2022 12:00:00 AM	\$315.00	Check		86	General Fund: \$185.00 Building Fund: \$130.00	
<input type="checkbox"/>	Decker, Ted	3/6/2022 12:00:00 AM	\$315.00	Check		87	General Fund: \$185.00 Building Fund: \$130.00	
<input type="checkbox"/>	Decker, Ted	4/6/2022 12:00:00 AM	\$315.00	Check		88	General Fund: \$185.00 Building Fund: \$130.00	
<input type="checkbox"/>	Decker, Ted	5/6/2022 12:00:00 AM	\$315.00	Check		89	General Fund: \$185.00 Building Fund: \$130.00	

50 500 5,000 4 Transactions





## Let's look at the Person Alias records

<b>Photo</b> ⚡	Gets or sets the <a href="#">BinaryFile</a> that contains the Person's photo.
<b>PhotoId</b> ⚡	Gets or sets the Id of the <a href="#">BinaryFile</a> that contains the photo of the Person.
<b>PhotoUrl</b> ⚡	Gets the URL of the person's photo.
<b>PreferredLanguageValue</b> ⚡	Gets or sets the <a href="#">DefinedValue</a> representing the Person's preferred language.
<b>PreferredLanguageValueId</b> ⚡	Gets or sets the <a href="#">DefinedValueId</a> of the <a href="#">DefinedValue</a> that represents the Preferred Language for this person.
<b>PrimaryAlias</b> ⚡	Gets the <a href="#">primary alias</a> .
<b>PrimaryAliasId</b> ⚡	Gets the <a href="#">primary alias</a> identifier.
<b>PrimaryCampus</b> ⚡	Gets or sets the person's <a href="#">primary campus</a> .
<b>PrimaryCampusId</b> ⚡	Gets or sets the campus id for the primary family. Note: This is computed on save, so any manual changes to this will be ignored.
<b>PrimaryFamily</b> ⚡	Gets or sets the <a href="#">primary family</a> .
<b>PrimaryFamilyId</b> ⚡	Gets or sets the group id for the <a href="#">PrimaryFamily</a> . Note: This is computed on save, so any manual changes to this will be ignored.
<b>PronunciationNote</b> ⚡	Gets or sets the notes for the pronunciation.



## Let's look at the Person Alias records

The screenshot shows a software interface with a sidebar and a main content area. The sidebar contains a list of items: Person viewed, Personal Device, Phone Number, and User Login. Below this is a 'Key' section with the following entries:

- A required field.
- A property on the database.
- Not mapped to the database.  
These fields are computed and are only available in the object.
- ⚡ These fields are available where Lava is supported.
- ✖ These methods or fields are obsolete and should not be used.

The main content area displays a list of properties for the 'Person' entity. A red arrow points from the 'Aliases' entry in the list to the corresponding entry in the 'Key' legend. The list includes:

- AdditionalLavaFields
- Age ⚡ Gets the Person's age.
- AgeBracket ⚡ Gets or sets the age bracket. This is a hard coded list of values defined in the code as an enumeration.  
[Show Values](#)
- AgeClassification ⚡ Gets or sets the age classification of the Person. Note: This is computed on save, so any manual changes to this will be ignored. This is a hard coded list of values defined in the code as an enumeration.  
[Show Values](#)
- AgePrecise ⚡ Gets the Person's precise age (includes the fraction of the year).
- Aliases ⚡ Gets or sets the aliases for this person.
- AllowsInteractiveBulkIndexing
- AnniversaryDate ⚡ Gets or sets the date of the Person's wedding anniversary. This property is nullable if the Person is not married or their anniversary date is not known.



# Person Aliases

## Person

Id: 5

FirstName: Theodore

NickName: Ted

LastName: Decker

## PersonAlias

Id: 16

PersonId: 5

...

## Financial Transaction

Id: 154

Date: 2/6/2022

PersonAliasId: 16

Id: 158

Date: 3/6/2022

PersonAliasId: 16



# Person Aliases

Person

Id: 5  
FirstName: Theodore  
NickName: Ted  
LastName: Decker

PersonAlias

Id: 16  
PersonId: 5  
...

Financial Transaction

Id: 154  
Date: 2/6/2022  
PersonAliasId: 16

Id: 158  
Date: 3/6/2022  
PersonAliasId: 16

Person

Id: 20  
FirstName: Teo  
NickName: Teo  
LastName: Decker

PersonAlias

Id: 29  
PersonId: 20  
...

Financial Transaction

Id: 161  
Date: 4/6/2022  
PersonAliasId: 29

Id: 163  
Date: 5/6/2022  
PersonAliasId: 29



# Person Aliases

Person

Id: 5  
FirstName: Theodore  
NickName: Ted  
LastName: Decker

PersonAlias

Id: 16  
PersonId: 5  
...

Financial Transaction

Id: 154  
Date: 2/6/2022  
PersonAliasId: 16

Id: 158  
Date: 3/6/2022  
PersonAliasId: 16

Person

Id: 20

PersonAlias

Id: 29  
PersonId: 5  
...

Financial Transaction

Id: 161  
Date: 4/6/2022  
PersonAliasId: 29

Id: 163  
Date: 5/6/2022  
PersonAliasId: 29



# Person Aliases

Person

Id: 5  
FirstName: Theodore  
NickName: Ted  
LastName: Decker

PersonAlias

Id: 16  
PersonId: 5  
...

Financial Transaction

Id: 154  
Date: 2/6/2022  
PersonAliasId: 16

Id: 158  
Date: 3/6/2022  
PersonAliasId: 16

PersonAlias

Id: 29  
PersonId: 5  
...

Financial Transaction

Id: 161  
Date: 4/6/2022  
PersonAliasId: 29

Id: 163  
Date: 5/6/2022  
PersonAliasId: 29



## Let's look at the Person Alias records

The screenshot shows a software interface with a sidebar and a main content area. The sidebar contains a list of items: Person viewed, Personal Device, Phone Number, and User Login. Below this is a 'Key' section with the following entries:

- A required field.
- A property on the database.
- Not mapped to the database.  
These fields are computed and are only available in the object.
- ⚡ These fields are available where Lava is supported.
- ✖ These methods or fields are obsolete and should not be used.

The main content area displays a list of properties for the 'Person' entity. A red arrow points from the 'Aliases' entry in the list to the corresponding entry in the 'Key' legend. The list includes:

- AdditionalLavaFields
- Age ⚡ Gets the Person's age.
- AgeBracket ⚡ Gets or sets the age bracket. This is a hard coded list of values defined in the code as an enumeration.  
[Show Values](#)
- AgeClassification ⚡ Gets or sets the age classification of the Person. Note: This is computed on save, so any manual changes to this will be ignored. This is a hard coded list of values defined in the code as an enumeration.  
[Show Values](#)
- AgePrecise ⚡ Gets the Person's precise age (includes the fraction of the year).
- Aliases ⚡ Gets or sets the aliases for this person.
- AllowsInteractiveBulkIndexing
- AnniversaryDate ⚡ Gets or sets the date of the Person's wedding anniversary. This property is nullable if the Person is not married or their anniversary date is not known.



# Person Aliases

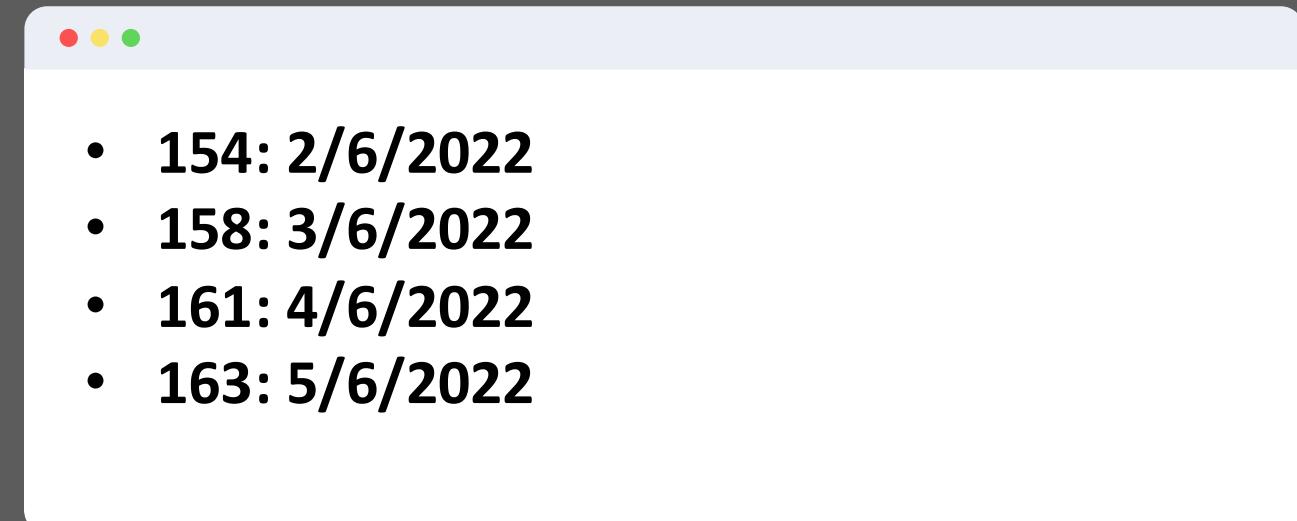
```
1 <ul>
2   {% for alias in CurrentPerson.Aliases %}
3     <li>
4       Alias Id: {{ alias.Id }}
5       <ul>
6         <li>Person Id: {{ alias.PersonId }}</li>
7       </ul>
8     </li>
9   {% endfor %}
10 </ul>
```





# Partial Solution

```
1  {% financialtransaction
2      where:'AuthorizedPersonAliasId == "17" || AuthorizedPersonAliasId == "29"'
3      iterator:'Transactions'
4      securityenabled:'false'
5  %}
6  {% for transaction in Transactions %}
7      <p>{{ transaction.Id }}: {{ transaction.TransactionDateTime | Date:'M/d/yyyy' }}</p>
8  {% endfor %}
9  {% endfinancialtransaction %}
```





# Expression

```
1  {% financialtransaction
2      expression:'AuthorizedPersonAlias.PersonId == "{{ CurrentPerson.Id }}"'
3      iterator:'Transactions'
4      securityenabled:'false'
5  %}
6  {% for transaction in Transactions %}
7      <p>{{ transaction.Id }}: {{ transaction.TransactionDateTime | Date:'M/d/yyyy' }}</p>
8  {% endfor %}
9  {% endfinancialtransaction %}
```



## Keep in mind:

The expression filters know the entity already, so don't type it.

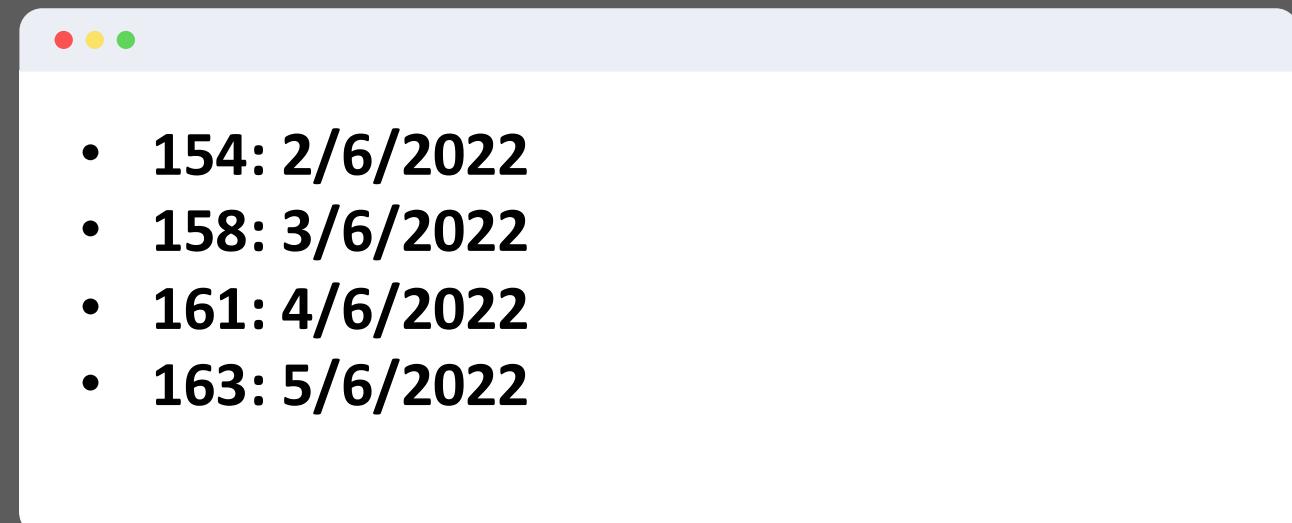


- **154: 2/6/2022**
- **158: 3/6/2022**
- **161: 4/6/2022**
- **163: 5/6/2022**



# Expression

```
1  {%
2      financialtransaction
3          expression:'AuthorizedPersonAlias.PersonId == "{{ CurrentPerson.Id }}"'
4          where:'TransactionDateTime > "2022-01-01" && TransactionDateTime < "2022-07-01"'
5          iterator:'Transactions'
6          securityenabled:'false'
7      %}
8      {%
9          for transaction in Transactions %
10             <p>{{ transaction.Id }}: {{ transaction.TransactionDateTime | Date:'M/d/yyyy' }}</p>
11      %}
12  {%
13      endfinancialtransaction %}
```





# But I Want The Transaction Amounts



# Expression

```
1  {% financialtransaction
2      expression:'AuthorizedPersonAlias.PersonId == "{{ CurrentPerson.Id }}"'
3      iterator:'Transactions'
4      securityenabled:'false'
5  %}
6  {% for transaction in Transactions %}
7      <p>{{ transaction.Id }}: {{ transaction.TransactionDateTime | Date:'M/d/yyyy' }}</p>
8  {% endfor %}
9  {% endfinancialtransaction %}
```



## Keep in mind:

The expression filters know the entity already, so don't type it.



- **154: 2/6/2022**
- **158: 3/6/2022**
- **161: 4/6/2022**
- **163: 5/6/2022**