

# Intelligent order management - Creating a provider and a Power Automate Flow

Dynamics 365 FastTrack  
Architecture Insights Series

Steven Koppens  
Senior FastTrack Solution Architect



---

# Agenda

- Mappings
- Creating a Provider
- Creating a Power Automate Flow
- Activating the Provider
- Testing the Order Intake Flow
- Resources

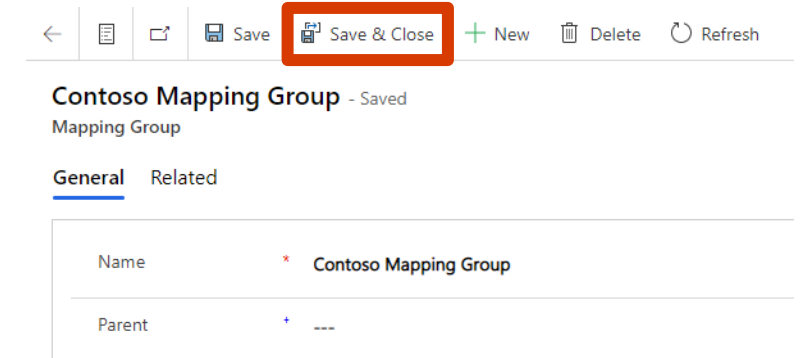
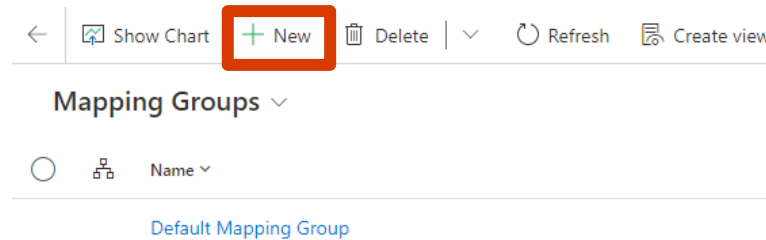
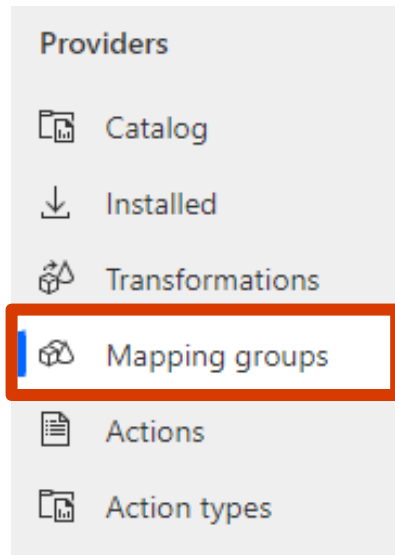


# Mappings



# Power Query and Mappings

- Mapping group



# Power Query and Mappings

- Setting up mappings

Change area

✓ Intelligent Order Ma...

Configurations

IO Intelligent Order ...

Mappings

Accounts

Contacts

Products

Orders

Order products

Fulfillment orders

Fulfillment order ...

Price lists

Warehouses

Currencies

Unit groups

Units

New Account Mapping - Saved

Account Mapping

General Related


Mapping Group \* Contoso Mapping Group

Account ---

Customer \* First Up Consultants

External Field Name \* ProviderName

External Field Value \* bob@FirstUpConsultants.com

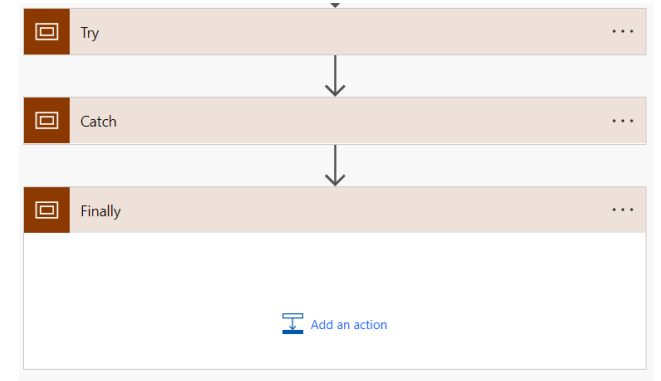
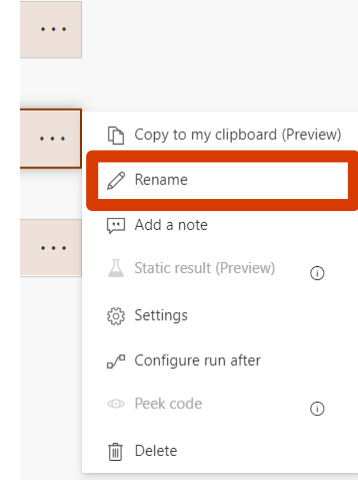
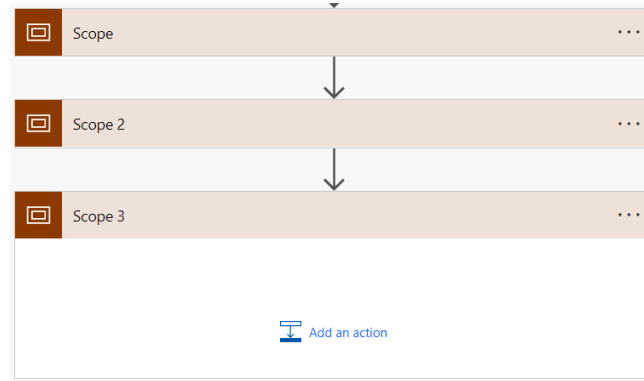
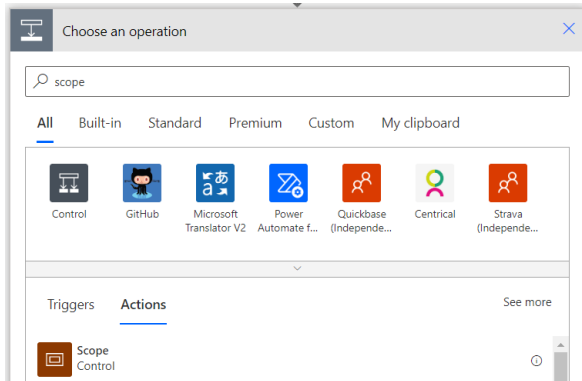


# Creating a Provider and a Power Automate Flow



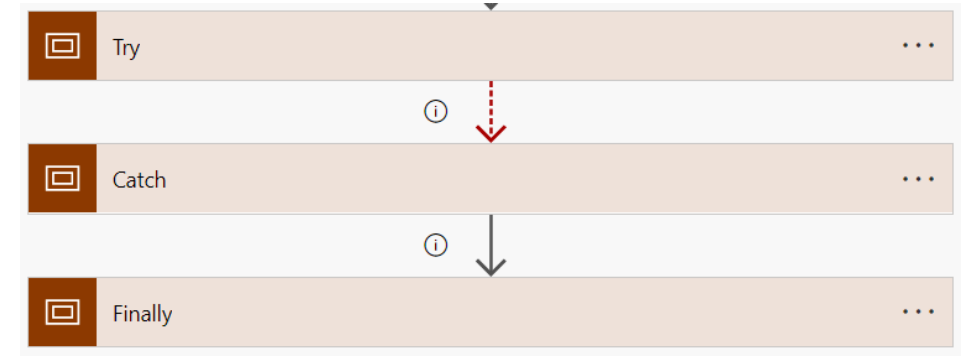
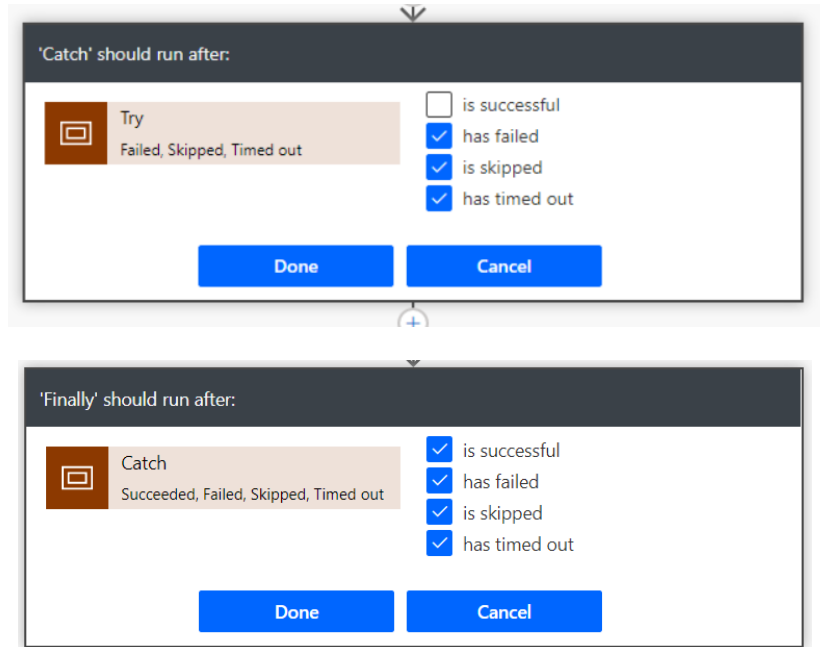
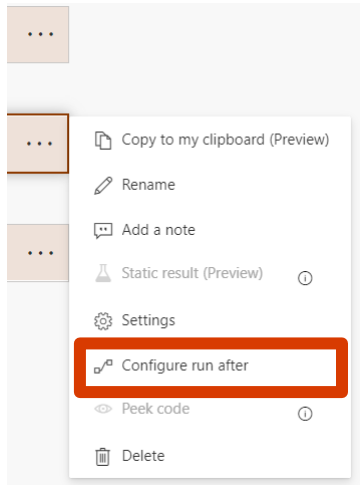
# Power Automation – Try, Catch, Finally Structure

- Try, Catch, Finally



# Power Automation – Try, Catch, Finally Structure

- Try, Catch, Finally





# Demo – Create an IOM Provider Definition



Microsoft Dynamics 365 for Operations Lab

Power Apps

Search

Environment  
D365 IOM SA LAB

SK

Home

Learn

Apps

Create

Dataverse

Flows

Chatbots

AI Builder

Solutions

New solution

Import solution

Open AppSource

Publish all customizations

Search

1 environment variable need to be updated.

Solutions

Publishers

History

| Display name   | Name                   | Created      | Version         | Managed | Publisher                | Solution check             |
|--|------------------------|--------------|-----------------|---------|--------------------------|----------------------------|
| Anchor solution for Dual Write Core                  | DualWriteCoreAnchor    | 6 months ago | 1.0.35          | Yes     | Dynamics 365             | Checked by publisher       |
| Anchor solution for Intelligent Order Manage...      | msdyn_IntelligentOr... | 1 year ago   | 1.0.0.4520      | Yes     | Dynamics 365             | Checked by publisher       |
| Anchor solution for Inventory Visibility applicat... | msdyn_InventorySer...  | 1 month ago  | 1.0.0.138       | Yes     | Dynamics 365             | Checked by publisher       |
| Common Data Services Default Solution                | Cr5baae                | 1 year ago   | 1.0.0.0         | No      | CDS Default Publisher    | Hasn't been run            |
| Contextual Help                                      | msdyn_ContextualH...   | 1 year ago   | 1.0.0.22        | Yes     | Dynamics 365             | Checked by publisher       |
| Contextual Help Base                                 | msdyn_ContextualH...   | 1 year ago   | 1.0.0.22        | Yes     | Dynamics 365             | Checked by publisher       |
| Crm Hub  | msdynce_CRMHub         | 1 month ago  | 9.0.22025.10001 | Yes     | Dynamics 365             | Checked by publisher       |
| Currency Exchange Rates                              | CurrencyExchangeRa...  | 6 months ago | 2.2.2.5         | Yes     | Dynamics 365             | Checked by publisher       |
| D365 Shared Controls                                 | msdynce_D365Share...   | 2 months ago | 9.0.220415.3    | Yes     | Dynamics 365             | Checked by publisher       |
| Default Solution                                     | Default                | 1 year ago   | 1.0             | No      | Default Publisher for... | Not supported for analysis |
| Distributed Order Management                         | DistributedOrderMa...  | 1 year ago   | 1.0.0.383       | Yes     | Dynamics 365             | Checked by publisher       |
| Dual Write Core                                      | msdyn_DualWriteCore    | 6 months ago | 1.0.35          | Yes     | MicrosoftCorporation     | Checked by publisher       |
| Dual-write applications core anchor                  | msdyn_DualWriteAp...   | 1 month ago  | 2.2.2.72        | Yes     | Dynamics 365             | Checked by publisher       |
| Dual-write applications core entity maps             | msdyn_DualWriteAp...   | 1 month ago  | 2.2.2.72        | Yes     | Dynamics 365             | Checked by publisher       |
| Dynamics 365 Company                                 | Dynamics365Compa...    | 6 months ago | 2.2.2.5         | Yes     | Microsoft Dynamics ...   | Checked by publisher       |

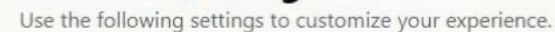
# Demo – Create a Power Automate Flow





# Demo – Add a Transformation and Power Automate Flow to the Provider





 [Manage](#)

 Manage

# Demo – Add a Provider



- Home
- Recent
- Pinned
- My work
- Getting started

## Insights

- Order Management
- Fulfillment
- System Monitoring
- Licenses
- Inventory Visibility

## Providers

- Catalog
- Installed
- Transformations
- Mapping groups
- Actions
- Action types

## Errors

- Provider inbound

## Orders

- Intelligent Order ...



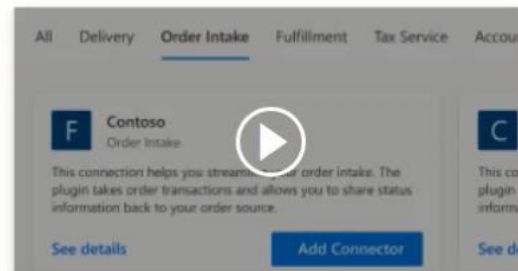
## Welcome, and get started

Use the following settings to customize your experience.



### Overview of Intelligent Order Management

Get an overview of the Dynamics 365 Intelligent Order Management Solution concept and how you can use that in your organization. [Learn more](#)

[→ Take a tour](#)[▶ Play video](#)

### Provider overview and configuration of 3rd party services

Get an overview of how to configure a provider with an external service provider. [Learn more](#)

[→ Take a tour](#)[▶ Play video](#)

### Design and orchestration flows for order fulfillment

Get an overview of the concepts for configuration orchestration flows. [Learn more](#)

[→ Take a tour](#)[▶ Play video](#)

## Get up and running

### Configure settings

Configure connections required by Intelligent Order Management application. [Learn more](#)

[Manage](#)

### Manage insights and dashboards

Unlock data driven KPI insights and configure analysis dashboards. [Learn more](#)

[Manage](#)



# Demo – Test the order intake provider



[+ New solution](#) [← Import solution](#) [📁 Open AppSource](#) [📄 Publish all customizations](#) ...

Search

1 environment variable need to be updated.

## Solutions

[Solutions](#) [Publishers](#) [History](#)

| Display name ↑ ↓                                     | Name ↓                 | Created ↓    | Version ↓       | Managed ↓ | Publisher ↓              | Solution check             |
|--|------------------------|--------------|-----------------|-----------|--------------------------|----------------------------|
| Anchor solution for Dual Write Core                  | DualWriteCoreAnchor    | 6 months ago | 1.0.35          | Yes       | Dynamics 365             | Checked by publisher       |
| Anchor solution for Intelligent Order Manage...      | msdyn_IntelligentOr... | 1 year ago   | 1.0.0.4520      | Yes       | Dynamics 365             | Checked by publisher       |
| Anchor solution for Inventory Visibility applicat... | msdyn_InventorySer...  | 1 month ago  | 1.0.0.138       | Yes       | Dynamics 365             | Checked by publisher       |
| Common Data Services Default Solution                | Cr5baae                | 1 year ago   | 1.0.0.0         | No        | CDS Default Publisher    | Hasn't been run            |
| Contextual Help                                      | msdyn_ContextualH...   | 1 year ago   | 1.0.0.22        | Yes       | Dynamics 365             | Checked by publisher       |
| Contextual Help Base                                 | msdyn_ContextualH...   | 1 year ago   | 1.0.0.22        | Yes       | Dynamics 365             | Checked by publisher       |
| Contoso IOM Flows                                    | ContosoIOMFlows        | 1 day ago    | 1.0.0.0         | No        | CDS Default Publisher    | Hasn't been run            |
| Contoso IOM Provider                                 | ContosoIOMProvider     | 3 days ago   | 1.0.0.0         | No        | CDS Default Publisher    | Hasn't been run            |
| Crm Hub  | msdynce_CRMHub         | 1 month ago  | 9.0.22025.10001 | Yes       | Dynamics 365             | Checked by publisher       |
| Currency Exchange Rates                              | CurrencyExchangeRa...  | 6 months ago | 2.2.2.5         | Yes       | Dynamics 365             | Checked by publisher       |
| D365 Shared Controls                                 | msdynce_D365Share...   | 2 months ago | 9.0.220415.3    | Yes       | Dynamics 365             | Checked by publisher       |
| Default Solution                                     | Default                | 1 year ago   | 1.0             | No        | Default Publisher for... | Not supported for analysis |
| Distributed Order Management                         | DistributedOrderMa...  | 1 year ago   | 1.0.0.383       | Yes       | Dynamics 365             | Checked by publisher       |
| Dual Write Core                                      | msdyn_DualWriteCore    | 6 months ago | 1.0.35          | Yes       | MicrosoftCorporation     | Checked by publisher       |
| Dual-write applications core anchor                  | msdyn_DualWriteAp...   | 1 month ago  | 2.2.2.72        | Yes       | Dynamics 365             | Checked by publisher       |

# Resources



# Resources - Mappings


## Accounts



New Account Mapping - Saved  
Account Mapping

General Related


|               |   |                      |                      |
|---------------|---|----------------------|----------------------|
| Mapping Group | +  Contoso Mapping Group | External Field Name  | * IOMAccountMapping  |
| Account       | ---   | External Field Value | * connie@contoso.com |
| Customer      | *  First Up Consultants  |                      |                      |






## Products

Active Product Mappings\* 

 Edit columns  Edit filters

Group By: 

(no grouping) 

| <input checked="" type="checkbox"/> Mapping Group  | Account  | Product  | External Field Name  | External Field Value  |
|---|---|---|---|--|
| Contoso Mapping Group   | ---   | Barista Home  | sku   | 883988211855   |
| Contoso Mapping Group   | ---   | Cleaning Kit  | sku   | cleankit443567   |

# Resources - Mappings

## Price lists

New Price List Mapping - Saved  
Price List Mapping

General Related

|               |   |                      |                       |
|---------------|---|----------------------|-----------------------|
| Mapping Group | +  Contoso Mapping Group | External Field Name  | * IOMPriceList        |
| Account       | ---   | External Field Value | * ContosoIOMPriceList |
| Price List    | *  Master price list     |                      |                       |

## Units

New Unit Mapping - Saved  
Unit Mapping

General Related

|               |   |                      |        |
|---------------|---|----------------------|--------|
| Mapping Group | +  Contoso Mapping Group | External Field Name  | * unit |
| Account       | ---   | External Field Value | * each |
| Unit          | *  ea                    |                      |        |



# Resources - Transformation

```
shared ImportMappingKey = [
  account = {
    [
      ExternalRecordKey = [IOMAccountMapping = Source[billingAddress][email]],
      SelectedFields = {"accountid"}
    ]
  },
  pricelevel = {
    [
      ExternalRecordKey = [IOMPriceList = "ContosoIOMPriceList"],
      SelectedFields = {"pricelevelid"}
    ]
  },
  product = List.Distinct(List.Transform(Source[orderdetails], each
    [
      ExternalRecordKey = [sku = _[sku]],
      SelectedFields = {"productid"}
    ])),
  uom = List.Distinct(List.Transform(Source[orderdetails], each
    [
      ExternalRecordKey = [unit = _[unit]],
      SelectedFields = {"uomid"}
    ]))
];
shared TransformSourceData =
let
  leadtime = 5,
  useWriteInProductEnabled = Source[useWriteInProductEnabled],
  isDualWriteEnabled = Source[isDualWriteEnabled],
  orderProducts = Source[orderdetails],
  billingAddress = Source[billingAddress],
  shippingAddress = Source[shippingAddress],
  account = IOM.MapRecord(IOM.MappingTables[account], [IOMAccountMapping = Source[billingAddress][email]]),
  pricelevel = IOM.MapRecord(IOM.MappingTables[pricelevel], [IOMPriceList = "ContosoIOMPriceList"]),

  orderheader = Record.FromTable
    (
      Table.SelectRows
        (
          Record.ToTable
            (
              [
                ordernumber = Text.From(Source[ordernumber]),
                name = ordernumber,
                #"customerid_account@odata.bind" = "/accounts(" & Text.From(account[accountid]) & ")",
                #"pricelevelid@odata.bind" = "/pricelevels(" & Text.From(pricelevel[pricelevelid]) & ")",
                billto_city = Record.FieldOrDefault(billingAddress, "billtocity"),
                billto_stateorprovince = Record.FieldOrDefault(billingAddress, "billtostateorprovince"),
                billto_country = Record.FieldOrDefault(billingAddress, "billtocountry"),
                billto_postalcode = Record.FieldOrDefault(billingAddress, "billtozip"),
                shipto_name = Record.FieldOrDefault(shippingAddress, "firstname") & " " & Record.FieldOrDefault(shippingAddress, "lastname"),
                shipto_contactname = shipto_name,
                shipto_line1 = Record.FieldOrDefault(shippingAddress, "street1"),
                shipto_line2 = Record.FieldOrDefault(shippingAddress, "street2"),
                shipto_city = Record.FieldOrDefault(shippingAddress, "shiptocity"),
                shipto_stateorprovince = Record.FieldOrDefault(shippingAddress, "shiptostateorprovince"),
                shipto_country = Record.FieldOrDefault(shippingAddress, "shiptocountry"),
                shipto_postalcode = Record.FieldOrDefault(shippingAddress, "shiptozip"),
                emailaddress = Record.FieldOrDefault(shippingAddress, "email")
              ]
            ), each [Value] <> null
          ),
        ), each [Value] <> null
    ),
  orderlines = List.Transform(orderProducts, each
    Record.FromTable
      (
        Table.SelectRows
          (
            Record.ToTable
              (
                [
                  productdescription = if useWriteInProductEnabled = true
                    then if Record.HasFields(IOM.MapRecord(IOM.MappingTables[product], [sku = Record.FieldOrDefault(_, "sku")])), "productid")
                    then null
                    else if isDualWriteEnabled = false
                    then Text.From(Record.FieldOrDefault(_, "sku"))
                    else null,
                    else null,
                  ispriceoverridden = true,
                  #"productid@odata.bind" = "/products(" & IOM.MapRecord(IOM.MappingTables[product], [sku = Record.FieldOrDefault(_, "sku")]))[productid] & ")",
                  #"uomid@odata.bind" = "/uoms(" & IOM.MapRecord(IOM.MappingTables[uom], [unit = Record.FieldOrDefault(_, "unit"])[uomid] & ")",
                  quantity = [quantity],
                  quantityshipped = Record.FieldOrDefault(_, "quantity_shipped"),
                  priceperunit = Decimal.From(Record.FieldOrDefault(_, "base_price")),
                  baseamount = Decimal.From(Record.FieldOrDefault(_, "base_total")),
                  tax = Decimal.From(Record.FieldOrDefault(_, "total_tax")),
                  shipto_name = Record.FieldOrDefault(orderheader, "shipto_name"),
                  shipto_contactname = Record.FieldOrDefault(orderheader, "shiptocontact_name"),
                  shipto_line1 = Record.FieldOrDefault(orderheader, "shipto_line1"),
                  shipto_line2 = Record.FieldOrDefault(orderheader, "shipto_line2"),
                  shipto_city = Record.FieldOrDefault(orderheader, "shipto_city"),
                  shipto_stateorprovince = Record.FieldOrDefault(orderheader, "shipto_stateorprovince"),
                  shipto_country = Record.FieldOrDefault(orderheader, "shipto_country"),
                  shipto_postalcode = Record.FieldOrDefault(orderheader, "shipto_postalcode")
                ]
              ), each [Value] <> null
            ),
          ), each [Value] <> null
        ),
      ),
    salesorder = Record.AddField(orderheader, "order_details", orderlines)
  in Text.FromBinary(Json.FromValue(salesorder));
```

```
shared ImportMappingKey = [
  account = {
    [
      ExternalRecordKey = [IOMAccountMapping = Source[billingAddress][email]],
      SelectedFields = {"accountid"}
    ]
  },
  pricelevel = {
    [
      ExternalRecordKey = [IOMPriceList = "ContosoIOMPriceList"],
      SelectedFields = {"pricelevelid"}
    ]
  },
  product = List.Distinct(List.Transform(Source[orderdetails], each
    [
      ExternalRecordKey = [sku = _[sku]],
      SelectedFields = {"productid"}
    ])),
  uom = List.Distinct(List.Transform(Source[orderdetails], each
    [
      ExternalRecordKey = [unit = _[unit]],
      SelectedFields = {"uomid"}
    ]))
];
shared TransformSourceData =
let
  leadtime = 5,
  useWriteInProductEnabled = Source[useWriteInProductEnabled],
  isDualWriteEnabled = Source[isDualWriteEnabled],
  orderProducts = Source[orderdetails],
  billingAddress = Source[billingAddress],
  shippingAddress = Source[shippingAddress],
  account = IOM.MapRecord(IOM.MappingTables[account], [IOMAccountMapping = Source[billingAddress][email]]),
  pricelevel = IOM.MapRecord(IOM.MappingTables[pricelevel], [IOMPriceList = "ContosoIOMPriceList"]),

  orderheader = Record.FromTable
    (
      Table.SelectRows
        (
          Record.ToTable
            (
              [
                ordernumber = Text.From(Source[ordernumber]),
                name = ordernumber,
                #"customerid_account@odata.bind" = "/accounts(" & Text.From(account[accountid]) & ")",
                #"pricelevelid@odata.bind" = "/pricelevels(" & Text.From(pricelevel[pricelevelid]) & ")",
                billto_city = Record.FieldOrDefault(billingAddress, "billtocity"),
                billto_stateorprovince = Record.FieldOrDefault(billingAddress, "billtostateorprovince"),
                billto_country = Record.FieldOrDefault(billingAddress, "billtocountry"),
                billto_postalcode = Record.FieldOrDefault(billingAddress, "billtozip"),
                shipto_name = Record.FieldOrDefault(shippingAddress, "firstname") & " " & Record.FieldOrDefault(shippingAddress, "lastname"),
                shipto_contactname = shipto_name,
                shipto_line1 = Record.FieldOrDefault(shippingAddress, "street1"),
                shipto_line2 = Record.FieldOrDefault(shippingAddress, "street2"),
                shipto_city = Record.FieldOrDefault(shippingAddress, "shiptocity"),
                shipto_stateorprovince = Record.FieldOrDefault(shippingAddress, "shiptostateorprovince"),
                shipto_country = Record.FieldOrDefault(shippingAddress, "shiptocountry"),
                shipto_postalcode = Record.FieldOrDefault(shippingAddress, "shiptozip"),
                emailaddress = Record.FieldOrDefault(shippingAddress, "email")
              ]
            ), each [Value] <> null
          ),
        ), each [Value] <> null
    ),
  orderlines = List.Transform(orderProducts, each
    Record.FromTable
      (
        Table.SelectRows
          (
            Record.ToTable
              (
                [
                  productdescription = if useWriteInProductEnabled = true
                    then if Record.HasFields(IOM.MapRecord(IOM.MappingTables[product], [sku = Record.FieldOrDefault(_, "sku")])), "productid")
                    then null
                    else if isDualWriteEnabled = false
                    then Text.From(Record.FieldOrDefault(_, "sku"))
                    else null,
                    else null,
                  ispriceoverridden = true,
                  #"productid@odata.bind" = "/products(" & IOM.MapRecord(IOM.MappingTables[product], [sku = Record.FieldOrDefault(_, "sku")]))[productid] & ")",
                  #"uomid@odata.bind" = "/uoms(" & IOM.MapRecord(IOM.MappingTables[uom], [unit = Record.FieldOrDefault(_, "unit"])[uomid] & ")",
                  quantity = [quantity],
                  quantityshipped = Record.FieldOrDefault(_, "quantity_shipped"),
                  priceperunit = Decimal.From(Record.FieldOrDefault(_, "base_price")),
                  baseamount = Decimal.From(Record.FieldOrDefault(_, "base_total")),
                  tax = Decimal.From(Record.FieldOrDefault(_, "total_tax")),
                  shipto_name = Record.FieldOrDefault(orderheader, "shipto_name"),
                  shipto_contactname = Record.FieldOrDefault(orderheader, "shiptocontact_name"),
                  shipto_line1 = Record.FieldOrDefault(orderheader, "shipto_line1"),
                  shipto_line2 = Record.FieldOrDefault(orderheader, "shipto_line2"),
                  shipto_city = Record.FieldOrDefault(orderheader, "shipto_city"),
                  shipto_stateorprovince = Record.FieldOrDefault(orderheader, "shipto_stateorprovince"),
                  shipto_country = Record.FieldOrDefault(orderheader, "shipto_country"),
                  shipto_postalcode = Record.FieldOrDefault(orderheader, "shipto_postalcode")
                ]
              ), each [Value] <> null
            ),
          ), each [Value] <> null
        ),
      ),
    salesorder = Record.AddField(orderheader, "order_details", orderlines)
  in Text.FromBinary(Json.FromValue(salesorder));
```



# Resources - Sample JSON Message

```
[
  {
    "useWriteInProductEnabled": true,
    "isDualWriteEnabled": false,
    "ordernumber": "ExternalContosoOrder025",
    "billingAddress":
    {
      "billtocity": "BELLEVUE",
      "billtostateorprovince": "WA",
      "billtocountry": "US",
      "billtozip": "98007",
      "email": "connie@contoso.com"
    },
    "shippingAddress":
    {
      "firstname": "Connie",
      "lastname": "Vrettos",
      "street1": "123 Coffee Street",
      "street2": "Suite 300",
      "shiptocity": "Redmond",
      "shiptostateorprovince": "WA",
      "shiptocountry": "US",
      "shiptozip": "98052",
      "email": "connie@contoso.com"
    },
    "orderdetails": [
      {
        "sku": "883988211855",
        "unit": "each",
        "quantity": 11,
        "base_price": 12.25,
        "base_total": 134.75,
        "total_tax": 20.21
      },
      {
        "sku": "cleankit443567",
        "unit": "each",
        "quantity": 21,
        "base_price": 20.00,
        "base_total": 420.00,
        "total_tax": 63.00
      }
    ]
  }
]
```



# Resources

- Intelligent Order Management -TechTalk
- [Dynamics 365 Intelligent Order Management | September 3, 2021 - Microsoft Dynamics Blog](#)
- Intelligent Order Management - Docs
- <https://docs.microsoft.com/en-us/dynamics365/intelligent-order-management/>
- Power Query
- <https://docs.microsoft.com/en-us/powerquery-m/>





Thank you



