**SOLUTION IMPLEMENTATION CHANGE**

1. **Requirement change development.**

**Attachment:**

“Regarding the technical implementation following is suggested:

1.      Have all Order Cubes (OI, TO and OB) in two versions

a.      First Version with Pre-calculated values ONLY, so always pre-calc., even if post-calc. is available

b.      Second Version with “most up to date” values, so take post-calc. if available and pre-calc. if there is no post-calc

i.     These Cubes should contain an additional Entity “Calculation Status” indicating whether the Order is “Pre-calc.” or “Post-calc.” (naming is just a suggestion)

So, establish a new Entity “Calculation Status” as well as building up all order data cubes in twice.”

1. **Clarification Requirement Change Development**
2. **Evaluation of Current System**
   1. Board
   2. DWH
      1. Context Diagram LoadToFact

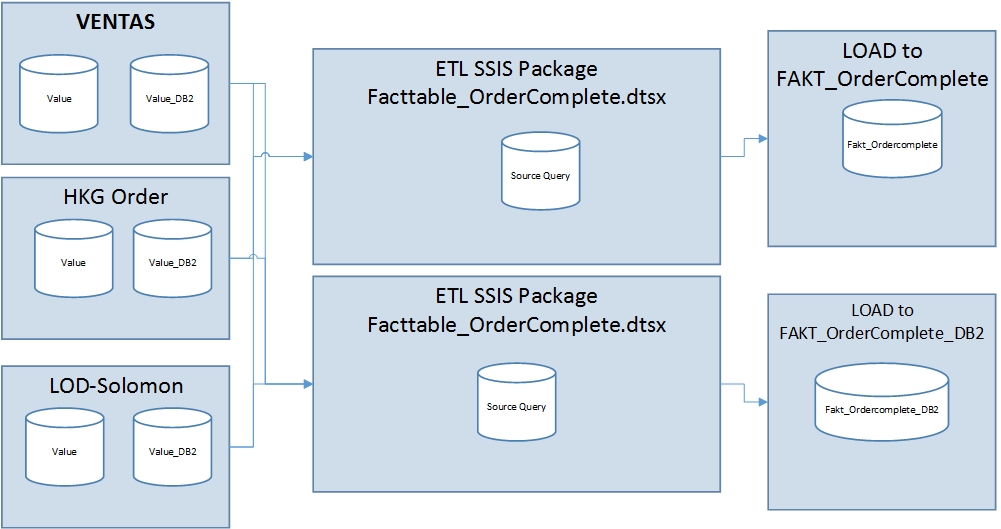


Figure 1: Diagram data flow Load to Fakt\_ordercomplete

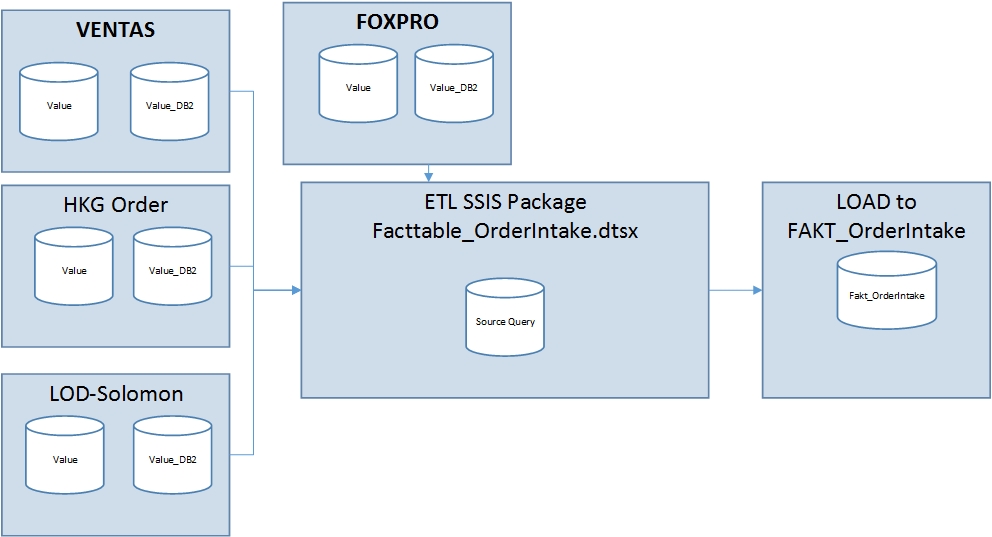


Figure 2: diagram data flow Fakt\_OrderIntake

* + 1. Structure Table and View for OrderComplete
* Table FAKT\_OrderComplete

[ordercomplete\_value] Field is stored the final value of Order

[ordercomplete\_value\_currency] Field is stored the final value\_currency of Order

[ordercomplete\_db2] Field

[ordercomplete\_db2\_currency] Field

* Current Mapping:
* **Ventas:Ventas\_Ordercomplete\_trade\_value**

Preis\_ges\_dm🡪ordercomplete\_value

Preis\_ges\_wr🡪ordercomplete\_value\_currency

Ignore🡪 ordercomplete\_db2

Ignore🡪 ordercomplete\_db2\_currency

* **Ventas:Ventas\_Ordercomplete\_commision\_Value**

Preis\_ges\_dm🡪ordercomplete\_value

Preis\_ges\_wr🡪ordercomplete\_value\_currency

Ignore🡪 ordercomplete\_db2

Ignore🡪 ordercomplete\_db2\_currency

* **LOD-Solomon: Non HKG orders and FOXPRO orders before 2013 - Fact Order Complete**

ordercomplete\_value 🡪ordercomplete\_value

ordercomplete\_value\_currency 🡪ordercomplete\_value\_currency

Ignore🡪 ordercomplete\_db2

Ignore🡪 ordercomplete\_db2\_currency

* **LOD-Solomon: HKG orders and FOXPRO orders since 2013 - Fact Order Complete**

ordercomplete\_value 🡪ordercomplete\_value

ordercomplete\_value\_currency 🡪ordercomplete\_value\_currency

Ignore🡪 ordercomplete\_db2

Ignore🡪 ordercomplete\_db2\_currency

* **Local-Orders: Local Orders\_OrderComplete\_Value**

Wert\_Wrg 🡪ordercomplete\_value

Ignore 🡪ordercomplete\_value\_currency

Ignore🡪 ordercomplete\_db2

Ignore🡪 ordercomplete\_db2\_currency

* Table FAKT\_OrderComplete\_DB2

[ordercomplete\_value] Field

[ordercomplete\_value\_currency] Field

[ordercomplete\_db2] Field is stored the final value\_db2 of order

[ordercomplete\_db3] Field is stored the final value\_db3 of order

[ordercomplete\_db2\_currency] Field is store the final value\_db2\_currency of order

* Mapping :
* **Ventas:Ventas\_Ordercomplete\_trade\_value**

Ignore 🡪ordercomplete\_value

Ignore 🡪ordercomplete\_value\_currency

DB2🡪 ordercomplete\_db2

Ignore🡪 ordercomplete\_db2\_currency

Ignore🡪 ordercomplete\_db3

* **Ventas:Ventas\_Ordercomplete\_commision\_Value**

Ignore 🡪ordercomplete\_value

Ignore 🡪ordercomplete\_value\_currency

DB2🡪 ordercomplete\_db2

Ignore🡪 ordercomplete\_db2\_currency

Ignore🡪 ordercomplete\_db3

* **LOD-solomon: Non HKG orders and FOXPRO orders before 2013 - Fact Order Complete**

Ignore 🡪ordercomplete\_value

Ignore 🡪ordercomplete\_value\_currency

ordercomplete\_db2🡪 ordercomplete\_db2

Ignore🡪 ordercomplete\_db2\_currency

ordercomplete\_db3🡪 ordercomplete\_db3

* **LOD-solomon: HKG orders and FOXPRO orders since 2013 - Fact Order Complete**

Ignore 🡪ordercomplete\_value

Ignore 🡪ordercomplete\_value\_currency

ordercomplete\_db2🡪 ordercomplete\_db2

Ignore🡪 ordercomplete\_db2\_currency

ordercomplete\_db3🡪 ordercomplete\_db3

* **Local-Orders: Local Orders\_OrderComplete\_Value**

Ignore 🡪ordercomplete\_value

Ignore 🡪ordercomplete\_value\_currency

DB2🡪 ordercomplete\_db2

Ignore🡪 ordercomplete\_db2\_currency

Ignore 🡪 ordercomplete\_db2

1. **Solution Development Change for Ordercomplete**
   1. Board
   2. DWH
      1. Staging Structure database.

Add new five columns into FAKT\_OrderComplete and FAKT\_OrderComplete\_DB2. Don’t change or replace any current columns to ensure that System will be run fine and avoid largely impacts cannot handle in a short time

* FAKT\_OrderComplete

[PreCal\_Value] [money] NULL,

[PreCal\_Value\_Currency] [money] NULL,

[PostCal\_Value] [money] NULL,

[PostCal\_Value\_Currency] [money] NULL,

[Calculation\_Status] [bit] NULL

* FAKT\_OrderComplete\_DB2

[PreCal\_Value\_DB2] [money] NULL,

[PreCal\_Value\_DB3] [money] NULL,

[PostCal\_Value\_DB2] [money] NULL,

[PostCal\_Value\_DB3] [money] NULL,

[Calculation\_Status] [bit] NULL

These views will be changed structure. In Fact, add new five columns in list of below views

* [dbo].[vTMP\_VENTAS\_OrderComplete\_Comm\_DB2pos]

In the first statement, we agreed that the Value DB2 is post-calculation. Just copying logic DB2 and then save as new columns Post\_Value\_DB2. Although this view has not contained PreCal\_Value\_DB2, PreCal\_Value\_DB3, PostCal\_Value\_DB3,[Calculation\_Status] before, we add these columns in this view

'0' AS PreCal\_Value\_DB2,

'0' AS PreCal\_Value\_DB3,

SUM(c.b\_betrag) \* CASE

WHEN b.preis\_ges\_dm\_ges = 0

THEN a.preis\_ges\_dm / (anz\_pos \* 1.0)

ELSE a.preis\_ges\_dm / b.preis\_ges\_dm\_ges \* 100

END / 100 AS PostCal\_Value\_DB2,

'0' AS PostCal\_Value\_DB3,

'1' AS [Calculation\_Status],

In the second statement, We agreed that Value DB2 is Pre-calculation. Just copying logic DB2 and then save as new column PreCal\_Value\_DB2. Although this view has not contained PostCal\_Value\_DB2, PostCal\_Value\_DB3, PreCal\_Value\_DB3,[Calculation\_Status] before, we add these columns in this view

b.db2 AS PreCal\_Value\_DB2,

'0' AS PreCal\_Value\_DB3,

'0' AS PostCal\_Value\_DB2,

'0' AS PostCal\_Value\_DB3,

'0' AS [Calculation\_Status],

* [dbo].[vTMP\_VENTAS\_OrderComplete\_Comm\_Value]

We agreed that with commission Order only have pre-calculation.Just copying bbo.LZ\_VENTAS\_akpos.preis\_ges\_wr, dbo.LZ\_VENTAS\_akpos.preis\_ges\_dm and them save as PreCal\_Value\_Currency and PreCal\_Value . Concurrently, add new columns PostCal\_Value, PostCal\_Value\_Currency, [Calculation\_Status]

* vTMP\_VENTAS\_OrderComplete\_Trade\_DB2pos

In the first statement , we agreed that the Value DB2 is post-calculation. Just copying logic DB2 and then save as new columns Post\_Value\_DB2 . Although this view has not contained PreCal\_Value\_DB2, PreCal\_Value\_DB3, PostCal\_Value\_DB3,[Calculation\_Status] before, we add these columns in this view

'0' AS PreCal\_Value\_DB2,

'0' AS PreCal\_Value\_DB3,

SUM(c.b\_betrag) \* CASE

WHEN b.preis\_ges\_dm\_ges = 0

THEN a.preis\_ges\_dm / (anz\_pos \* 1.0)

ELSE a.preis\_ges\_dm / b.preis\_ges\_dm\_ges \* 100

END / 100 AS PostCal\_Value\_DB2,

'0' AS PostCal\_Value\_DB3,

'1' AS [Calculation\_Status],

In the second statement, We agreed that Value DB2 is Pre-calculation. Just copying logic DB2 and then save as new column PreCal\_Value\_DB2. Although this view has not contained PostCal\_Value\_DB2, PostCal\_Value\_DB3, PreCal\_Value\_DB3,[Calculation\_Status] before, we add these columns in this view

b.db2 AS PreCal\_Value\_DB2,

'0' AS PreCal\_Value\_DB3,

'0' AS PostCal\_Value\_DB2,

'0' AS PostCal\_Value\_DB3,

'0' AS [Calculation\_Status],

* vTMP\_VENTAS\_OrderComplete\_Trade\_Value

Don’t change logic calculation of two fields because two fields is used in another views so that we add new logic calculation to create new fields(PreCal\_Value ,PreCal\_Value\_Currency , PostCal\_Value\_Currency , PostCal\_Value, Calculation\_Status ) which rely on two fields existing (preis\_ges\_wr, preis\_ges\_dm)

a.preis\_ges\_wr AS PreCal\_Value\_Currency,

a.preis\_ges\_dm AS PreCal\_Value,

CASE

WHEN a.abgerechnet\_dat IS NOT NULL

THEN b.preis\_ges\_wr

ELSE 0

END AS PostCal\_Value\_Currency,

CASE

WHEN a.abgerechnet\_dat IS NOT NULL

THEN b.preis\_ges\_dm

ELSE 0

END AS PostCal\_Value,

CASE

WHEN a.abgerechnet\_dat IS NOT NULL

THEN 1

ELSE 0

END AS [Calculation\_Status],

-- if abgrechnet\_dat is null then order intake value else invoice values

CASE

WHEN a.abgerechnet\_dat IS NULL

THEN a.preis\_ges\_wr

ELSE b.preis\_ges\_wr

END AS preis\_ges\_wr,

CASE

WHEN a.abgerechnet\_dat IS NULL

THEN a.preis\_ges\_dm

ELSE b.preis\_ges\_dm

END AS preis\_ges\_dm,

* + 1. SSIS Package ETL

Change source query script to load data into FAKT\_OrderComplete and edit mapping from source to destination of FAKT\_OrderComplete\_DB2

Source query FAKT\_Complete

PreCal\_Value 🡪 PreCal\_Value

PreCal\_Value\_Currency 🡪 PreCal\_Value\_Currency

PostCal\_Value 🡪 PostCal\_Value

PostCal\_Value\_Currency 🡪 PostCal\_Value\_Currency

Calculation\_Status 🡪 Calculation\_Status

Change source query script to load data into FAKT\_OrderComplete and edit mapping from source to destination of FAKT\_OrderComplete\_DB2

Source query FAKT\_Complete \_DB2

PreCal\_Value\_DB2, 🡪 PreCal\_Value\_DB2,

PreCal\_Value\_DB3 🡪 PreCal\_Value\_DB3

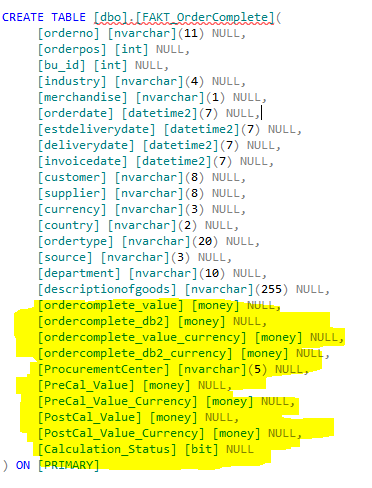
PostCal\_Value\_DB2 🡪 PostCal\_Value\_DB2

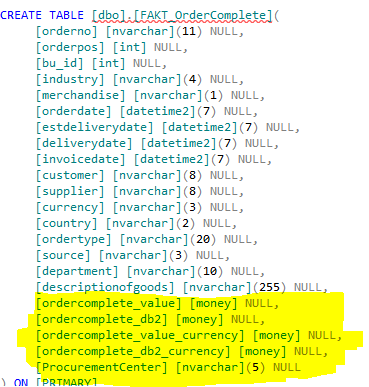
PostCal\_Value\_DB3 🡪 PostCal\_Value\_DB3

Calculation\_Status 🡪 Calculation\_Status

1. Implementation for Ordercomplete
   1. DWH
      1. Staging Structure Database

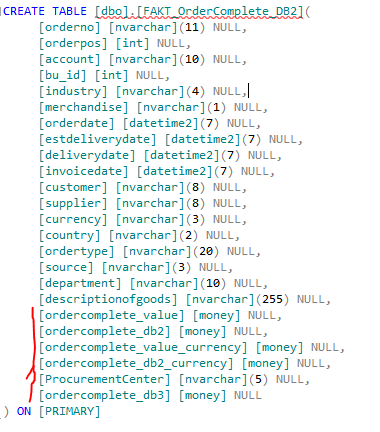
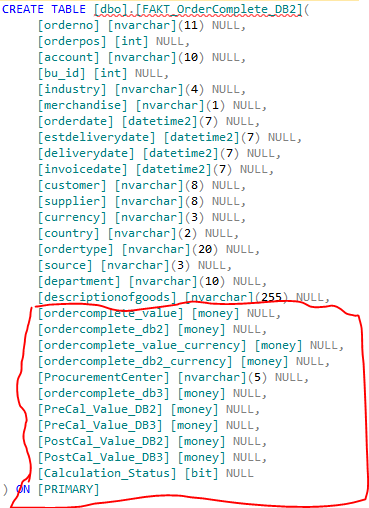
* Change Structure FAKT\_OrderComplete table by adding new five columns

 Old structure New structure



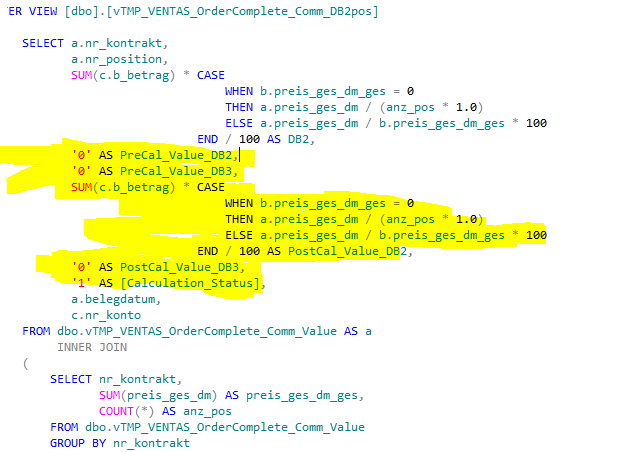
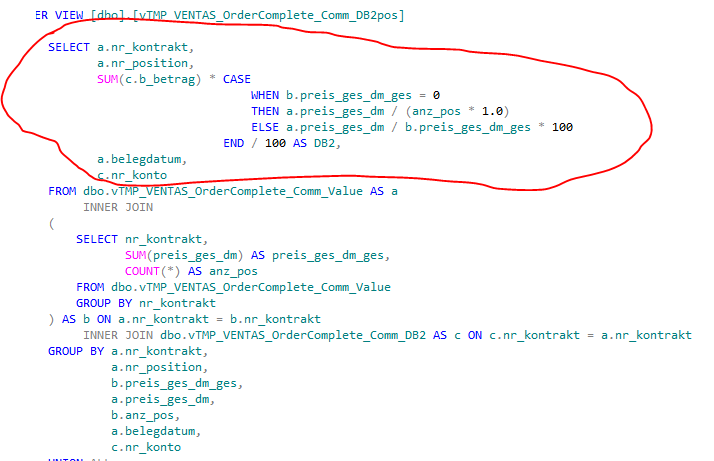
* Change structure FAKT\_OrderComplete\_DB2 table by adding new five columns

Old Structure New Structure



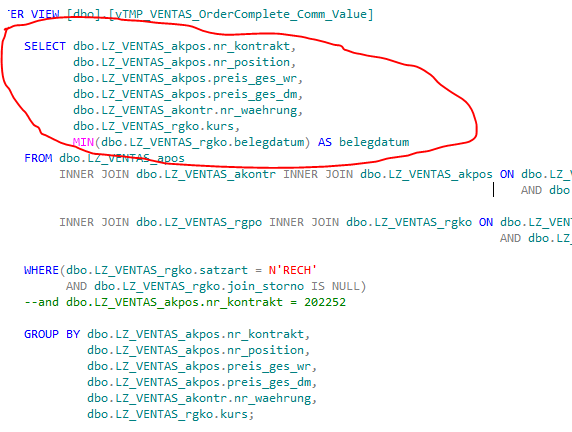
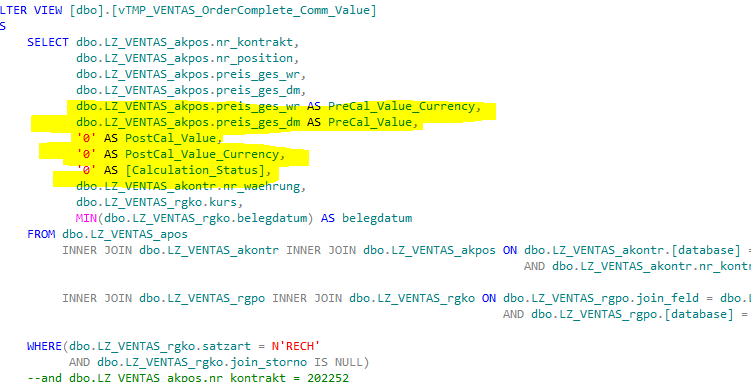
* Change Structure view [vTMP\_VENTAS\_OrderComplete\_Comm\_DB2pos]

Old Structure New Structure



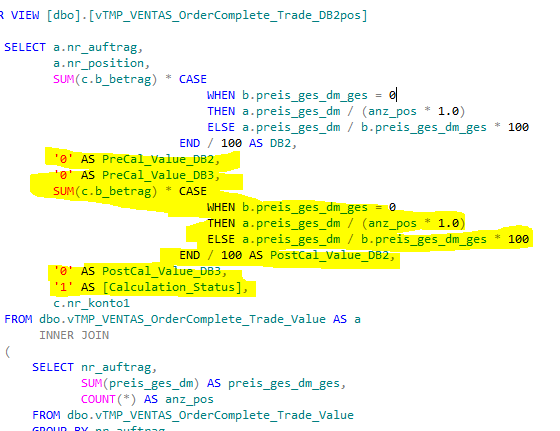
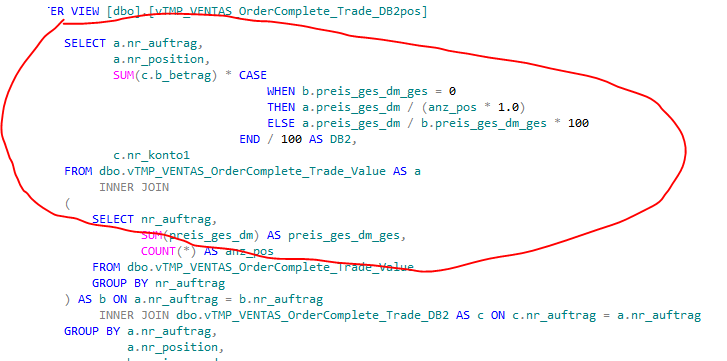
* Change structure view [vTMP\_VENTAS\_OrderComplete\_Comm\_Value]

Old Structure New Structure



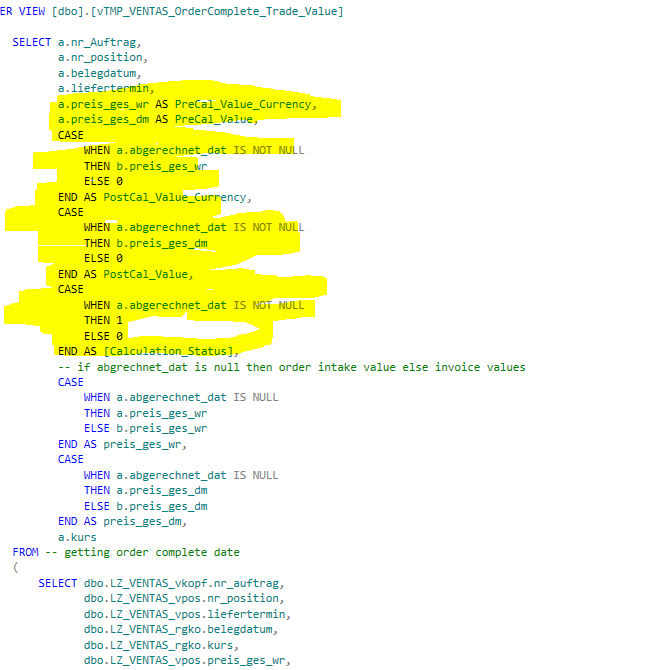
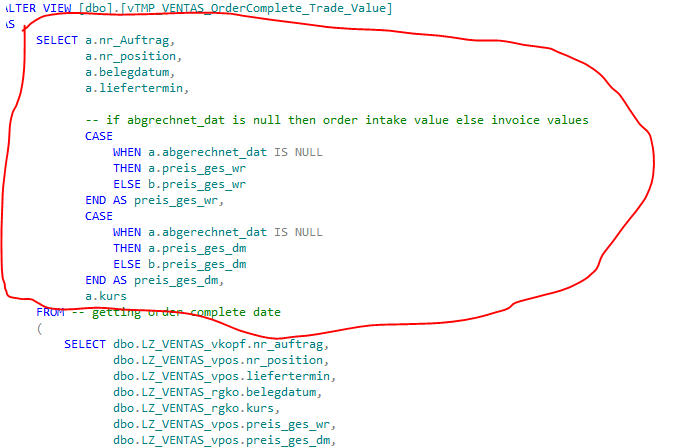
* Change structure view [vTMP\_VENTAS\_OrderComplete\_Trade\_DB2pos]

Old Structure New Structure



* Change Structure View [vTMP\_VENTAS\_OrderComplete\_Trade\_Value]

Old Structure New Structure



5.1.2. SSIS Package

- Change Source Query Load to Fact from Ventas, Local HKG Orders, LOD-Solomon to FAKT\_OrderComplete table and FAKT\_OrderComplete\_DB2

Old Script Source Query New Script Source Query

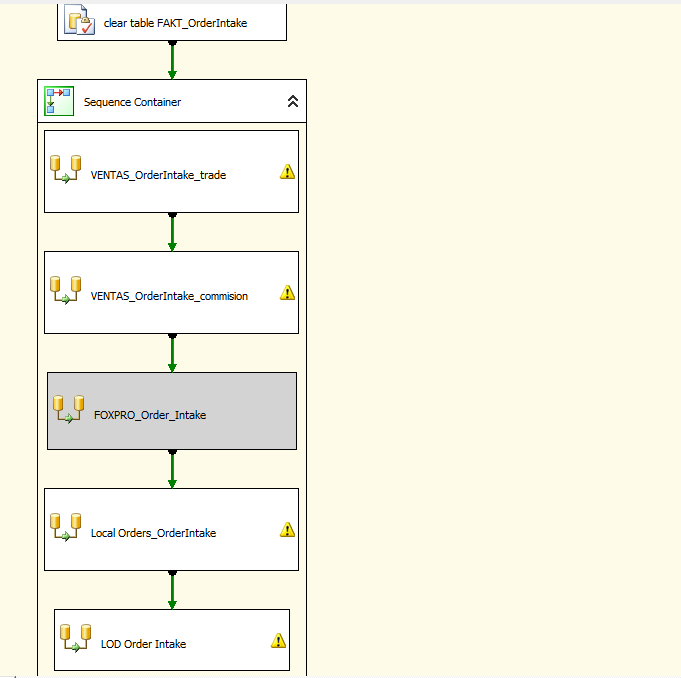
 

* 1. Board

1. Solution and Implementation change for OrderIntake

**6.1. SSIS package**

Control data flow



**6.1.1. Ventas source Trade**

**Original source query load data**

SELECT a.nr\_auftrag,

a.erfasst,

a.abgerechnet\_dat,

a.betreff,

a.nr\_kunde,

a.nr\_liefnt,

a.bez,

a.nr\_gart AS industry,

a.cty, a.department,

a.nr\_position\_vpos,

a.liefertermin,

RTRIM(a.Merchandise) AS Merchandise,

a.preis\_ges\_dm,

a.preis\_ges\_wr,

b.DB2,

a.bu\_id,

a.currency,

a.Backlogrelevant,

a.office\_reference

FROM dbo.vTMP\_VENTAS\_OrderIntake\_Trade\_02\_BU AS a INNER JOIN

dbo.vTMP\_VENTAS\_OrderIntake\_Trade\_03\_DB2pos AS b ON a.nr\_auftrag = b.nr\_auftrag

AND a.nr\_position\_vpos = b.nr\_position\_vpos

WHERE (a.erfasst >= '2008-01-01')

**Mapping:**

preis\_ges\_dm 🡪 orderintake\_value

preis\_ges\_wr 🡪 orderintake\_value\_currency

DB2 🡪 orderintake\_db2

**Current Logic data :get all value Pre of all orderno**

**Change logic to get post value of all order had post value.**

/\*VENTAS\_OrderIntake\_trade\*/

WITH temp1(nr\_auftrag,erfasst,abgerechnet\_dat,betreff,nr\_kunde,nr\_liefnt,bez,industry,cty,department,nr\_position\_vpos,liefertermin,Merchandise,preis\_ges\_dm,preis\_ges\_wr,DB2,bu\_id,currency,Backlogrelevant,office\_reference)

--temp1 table stores all data value Pre of All orderno. this is original source

AS

(SELECT a.nr\_auftrag,

a.erfasst,

a.abgerechnet\_dat,

a.betreff,

a.nr\_kunde,

a.nr\_liefnt,

a.bez,

a.nr\_gart AS industry,

a.cty,

a.department,

a.nr\_position\_vpos,

a.liefertermin,

RTRIM(a.Merchandise) AS Merchandise,

a.preis\_ges\_dm,

a.preis\_ges\_wr,

b.DB2,

a.bu\_id,

a.currency,

a.Backlogrelevant,

a.office\_reference

FROM dbo.vTMP\_VENTAS\_OrderIntake\_Trade\_02\_BU AS a

INNER JOIN dbo.vTMP\_VENTAS\_OrderIntake\_Trade\_03\_DB2pos AS b ON a.nr\_auftrag = b.nr\_auftrag

AND a.nr\_position\_vpos = b.nr\_position\_vpos

WHERE(a.erfasst >= '2008-01-01')

),

--because temp1 table stores all data value Pre of All orderno. it contains Pre value of orders not yet complete and complete. we find Ordercompletes to get post Value. So that we create temp2 to include Pre value and Post value

temp2(nr\_auftrag,erfasst,abgerechnet\_dat,betreff,nr\_kunde,nr\_liefnt,bez,industry,cty,department,nr\_position\_vpos,liefertermin,Merchandise,preis\_ges\_dm,preis\_ges\_wr,DB2,bu\_id,currency,Backlogrelevant,office\_reference,Postcal\_IntakeDB2

,Postcal\_IntakeDB2\_currency,PostCal\_Value\_DB3,Precal\_IntakeDB2,Precal\_IntakeDB2\_Currency,Precal\_IntakeDB3,Postcal\_IntakeDB3,

Precal\_IntakeValue\_Currency,Precal\_IntakeValue,Postcal\_IntakeValue\_Currency,Postcal\_IntakeValue,Calculation\_Status)

AS

(SELECT FAKT\_OrderIntake.nr\_auftrag,

FAKT\_OrderIntake.erfasst,

FAKT\_OrderIntake.abgerechnet\_dat,

FAKT\_OrderIntake.betreff,

FAKT\_OrderIntake.nr\_kunde,

FAKT\_OrderIntake.nr\_liefnt,

FAKT\_OrderIntake.bez,

FAKT\_OrderIntake.industry,

FAKT\_OrderIntake.cty,

FAKT\_OrderIntake.department,

FAKT\_OrderIntake.nr\_position\_vpos,

FAKT\_OrderIntake.liefertermin,

FAKT\_OrderIntake.Merchandise,

FAKT\_OrderIntake.preis\_ges\_dm,

FAKT\_OrderIntake.preis\_ges\_wr,

FAKT\_OrderIntake.DB2,

FAKT\_OrderIntake.bu\_id,

FAKT\_OrderIntake.currency,

FAKT\_OrderIntake.Backlogrelevant,

FAKT\_OrderIntake.office\_reference,

vTMP\_VENTAS\_OrderComplete\_Trade\_DB2pos.Postcal\_IntakeDB2,

'0' AS Postcal\_IntakeDB2\_currency,

vTMP\_VENTAS\_OrderComplete\_Trade\_DB2pos.PostCal\_Value\_DB3,

vTMP\_VENTAS\_OrderComplete\_Trade\_DB2pos.Precal\_IntakeDB2,

'0' AS Precal\_IntakeDB2\_Currency,

vTMP\_VENTAS\_OrderComplete\_Trade\_DB2pos.Precal\_IntakeDB3,

'0' AS Postcal\_IntakeDB3,

dbo.vTMP\_VENTAS\_OrderComplete\_Trade\_Value.PreCal\_Value\_Currency AS Precal\_IntakeValue\_Currency,

dbo.vTMP\_VENTAS\_OrderComplete\_Trade\_Value.PreCal\_Value AS Precal\_IntakeValue,

dbo.vTMP\_VENTAS\_OrderComplete\_Trade\_Value.PostCal\_Value\_Currency AS Postcal\_IntakeValue\_Currency,

dbo.vTMP\_VENTAS\_OrderComplete\_Trade\_Value.PostCal\_Value AS Postcal\_IntakeValue,

dbo.vTMP\_VENTAS\_OrderComplete\_Trade\_Value.[Calculation\_Status] AS Calculation\_Status

FROM temp1 AS FAKT\_OrderIntake

--Get Post Value And Pre value of orderscomplete from vTMP\_VENTAS\_OrderComplete\_Trade\_Value

LEFT JOIN dbo.vTMP\_VENTAS\_OrderComplete\_Trade\_Value ON dbo.vTMP\_VENTAS\_OrderComplete\_Trade\_Value.nr\_auftrag = FAKT\_OrderIntake.nr\_auftrag

AND dbo.vTMP\_VENTAS\_OrderComplete\_Trade\_Value.nr\_position = FAKT\_OrderIntake.nr\_position\_vpos

LEFT JOIN

--Get Post value DB2 and Pre Value DB2 of orderscomplete from vTMP\_VENTAS\_OrderComplete\_Trade\_DB2pos

(

SELECT dbo.vTMP\_VENTAS\_OrderComplete\_Trade\_DB2pos.nr\_auftrag,

dbo.vTMP\_VENTAS\_OrderComplete\_Trade\_DB2pos.nr\_position,

SUM(CASE

WHEN dbo.vTMP\_VENTAS\_OrderComplete\_Trade\_DB2pos.nr\_konto1 = '3730'

AND dbo.vTMP\_VENTAS\_OrderComplete\_Trade\_DB2pos.[Calculation\_Status] = 1

THEN dbo.vTMP\_VENTAS\_OrderComplete\_Trade\_DB2pos.PostCal\_Value\_DB2 / 2

ELSE dbo.vTMP\_VENTAS\_OrderComplete\_Trade\_DB2pos.PostCal\_Value\_DB2

END) AS Postcal\_IntakeDB2,

SUM(CAST(dbo.vTMP\_VENTAS\_OrderComplete\_Trade\_DB2pos.PostCal\_Value\_DB3 AS INT)) AS PostCal\_Value\_DB3,

SUM(CASE

WHEN dbo.vTMP\_VENTAS\_OrderComplete\_Trade\_DB2pos.nr\_konto1 = '3730'

AND dbo.vTMP\_VENTAS\_OrderComplete\_Trade\_DB2pos.[Calculation\_Status] = 0

THEN dbo.vTMP\_VENTAS\_OrderComplete\_Trade\_DB2pos.PreCal\_Value\_DB2 / 2

ELSE dbo.vTMP\_VENTAS\_OrderComplete\_Trade\_DB2pos.PreCal\_Value\_DB2

END) AS Precal\_IntakeDB2,

SUM(CAST(dbo.vTMP\_VENTAS\_OrderComplete\_Trade\_DB2pos.PreCal\_Value\_DB3 AS INT)) AS Precal\_IntakeDB3

FROM dbo.vTMP\_VENTAS\_OrderComplete\_Trade\_DB2pos

GROUP BY dbo.vTMP\_VENTAS\_OrderComplete\_Trade\_DB2pos.nr\_auftrag,

dbo.vTMP\_VENTAS\_OrderComplete\_Trade\_DB2pos.nr\_position

) AS vTMP\_VENTAS\_OrderComplete\_Trade\_DB2pos ON vTMP\_VENTAS\_OrderComplete\_Trade\_DB2pos.nr\_auftrag = FAKT\_OrderIntake.nr\_auftrag

AND vTMP\_VENTAS\_OrderComplete\_Trade\_DB2pos.nr\_position = FAKT\_OrderIntake.nr\_position\_vpos)

-- this is the process to solve null value. if null value then 0 from temp2

SELECT a.nr\_auftrag,

a.erfasst,

a.abgerechnet\_dat,

a.betreff,

a.nr\_kunde,

a.nr\_liefnt,

a.bez,

a.industry,

a.cty,

a.department,

a.nr\_position\_vpos,

a.liefertermin,

RTRIM(a.Merchandise) AS Merchandise,

a.preis\_ges\_dm,

a.preis\_ges\_wr,

a.DB2,

a.bu\_id,

a.currency,

a.Backlogrelevant,

a.office\_reference,

CASE

WHEN ISNULL(a.Precal\_IntakeValue, 0) = 0

THEN a.preis\_ges\_dm

ELSE ISNULL(a.Precal\_IntakeValue, 0)

END AS Precal\_IntakeValue,

CASE

WHEN ISNULL(a.Precal\_IntakeDB2, 0) = 0

THEN a.DB2

ELSE ISNULL(a.Precal\_IntakeDB2, 0)

END AS Precal\_IntakeDB2,

ISNULL(a.Precal\_IntakeDB2\_Currency, 0) AS Precal\_IntakeDB2\_Currency,

CASE

WHEN ISNULL(a.Precal\_IntakeValue\_Currency, 0) = 0

THEN a.preis\_ges\_wr

ELSE ISNULL(a.Precal\_IntakeValue\_Currency, 0)

END AS Precal\_IntakeValue\_Currency,

ISNULL(a.Postcal\_IntakeValue, 0) AS Postcal\_IntakeValue,

ISNULL(a.Postcal\_IntakeValue\_Currency, 0) AS Postcal\_IntakeValue\_Currency,

ISNULL(a.Postcal\_IntakeDB2, 0) AS Postcal\_IntakeDB2,

ISNULL(a.Postcal\_IntakeDB2\_currency, 0) AS Postcal\_IntakeDB2\_currency,

ISNULL(a.Precal\_IntakeDB3, 0) AS Precal\_IntakeDB3,

ISNULL(a.Postcal\_IntakeDB3, 0) AS Postcal\_IntakeDB3,

CASE

WHEN a.[Calculation\_Status] IS NULL

THEN 0

ELSE a.[Calculation\_Status]

END AS Calculation\_Status

FROM temp2 AS a;

**New Mapping :**

Precal\_IntakeValue 🡪Precal\_IntakeValue,

Precal\_IntakeDB2🡪Precal\_IntakeDB2,

Precal\_IntakeDB2\_Currency 🡪Precal\_IntakeDB2\_Currency,

Precal\_IntakeValue\_Currency 🡪Precal\_IntakeValue\_Currency,

Postcal\_IntakeValue 🡪Postcal\_IntakeValue,

Postcal\_IntakeValue\_Currency 🡪Postcal\_IntakeValue\_Currency,

Postcal\_IntakeDB2🡪Postcal\_IntakeDB2,

Postcal\_IntakeDB2\_currency 🡪Postcal\_IntakeDB2\_currency,

Precal\_IntakeDB3🡪Precal\_IntakeDB3,

Postcal\_IntakeDB3🡪Postcal\_IntakeDB3,

[Calculation\_Status]🡪[Calculation\_Status]

### 6.1.2. Ventas source commission

**Original Source Query Load**

SELECT b.nr\_position,

a.nr\_kontrakt,

a.abgerechnet\_dat,

a.nr\_liefnt,

a.nr\_kunde,

a.nr\_gart AS industry,

a.erfasst,

a.seriell,

a.liefertermin,

a.bez,

RTRIM(a.Merchandise) AS Merchandise,

a.nr\_adress\_k,

a.cty,

a.department,

a.preis\_ges\_dm,

a.preis\_ges\_wr,

b.DB2,

a.bu\_id,

a.currency,

a.Backlogrelevant,

a.office\_reference

FROM dbo.vTMP\_VENTAS\_OrderIntake\_Comm\_02\_BU AS a

INNER JOIN dbo.vTMP\_VENTAS\_OrderIntake\_Comm\_03\_DB2pos AS b ON a.nr\_kontrakt = b.nr\_kontrakt

AND a.nr\_position = b.nr\_position

WHERE(a.erfasst >= '2008-01-01');

**Mapping:**

preis\_ges\_dm 🡪 orderintake\_value

preis\_ges\_wr 🡪 orderintake\_value\_currency

DB2 🡪 orderintake\_db2

**Current Logic data: get all value Pre of all orderno**

**Change logic to get post value of all order had post value.**

/\*VENTAS\_OrderIntake\_commision\*/

GO

IF OBJECT\_ID('dbo.vTMPSourceLoadOderIntakeDB2\_Post', 'V') IS NOT NULL

DROP VIEW [dbo].[vTMPSourceLoadOderIntakeDB2\_Post]

GO

--Because with Ventas Commission order doesn't not have post value for ordercomplete\_value, only pre value for orders. we can get post value DB2 and is not with Post Value

--vTMPSourceLoadOderIntakeDB2\_Post will contain Post Value DB2 of ordercompletes

CREATE VIEW [dbo].[vTMPSourceLoadOderIntakeDB2\_Post]

AS

SELECT dbo.vTMP\_VENTAS\_OrderComplete\_Comm\_DB2pos.nr\_kontrakt,

dbo.vTMP\_VENTAS\_OrderComplete\_Comm\_DB2pos.nr\_position,

dbo.vTMP\_VENTAS\_OrderComplete\_Comm\_DB2pos.Calculation\_Status,

SUM(CASE

WHEN dbo.vTMP\_VENTAS\_OrderComplete\_Comm\_DB2pos.nr\_konto = '3730'

AND dbo.vTMP\_VENTAS\_OrderComplete\_Comm\_DB2pos.[Calculation\_Status] = 0

THEN dbo.vTMP\_VENTAS\_OrderComplete\_Comm\_DB2pos.PreCal\_Value\_DB2 / 2

ELSE dbo.vTMP\_VENTAS\_OrderComplete\_Comm\_DB2pos.PreCal\_Value\_DB2

END) AS Precal\_IntakeDB2,

SUM(CAST(dbo.vTMP\_VENTAS\_OrderComplete\_Comm\_DB2pos.PreCal\_Value\_DB3 AS INT)) AS Precal\_IntakeDB3,

SUM(CASE

WHEN dbo.vTMP\_VENTAS\_OrderComplete\_Comm\_DB2pos.nr\_konto = '3730'

AND dbo.vTMP\_VENTAS\_OrderComplete\_Comm\_DB2pos.[Calculation\_Status] = 1

THEN dbo.vTMP\_VENTAS\_OrderComplete\_Comm\_DB2pos.PostCal\_Value\_DB2 / 2

ELSE dbo.vTMP\_VENTAS\_OrderComplete\_Comm\_DB2pos.PostCal\_Value\_DB2

END) AS Postcal\_IntakeDB2,

SUM(CAST(dbo.vTMP\_VENTAS\_OrderComplete\_Comm\_DB2pos.PostCal\_Value\_DB3 AS INT)) AS Postcal\_IntakeDB3

FROM dbo.vTMP\_VENTAS\_OrderComplete\_Comm\_DB2pos

GROUP BY dbo.vTMP\_VENTAS\_OrderComplete\_Comm\_DB2pos.nr\_kontrakt,

dbo.vTMP\_VENTAS\_OrderComplete\_Comm\_DB2pos.nr\_position,

dbo.vTMP\_VENTAS\_OrderComplete\_Comm\_DB2pos.Calculation\_Status

GO

GO

IF OBJECT\_ID('dbo.vTMPSourceLoadOderIntake', 'V') IS NOT NULL

DROP VIEW [dbo].[vTMPSourceLoadOderIntake]

GO

--vTMPSourceLoadOderIntake is Original source, get all Pre value, Pre value DB2 of all orders

CREATE VIEW [dbo].[vTMPSourceLoadOderIntake]

AS

SELECT b.nr\_position,

a.nr\_kontrakt,

a.abgerechnet\_dat,

a.nr\_liefnt,

a.nr\_kunde,

a.nr\_gart AS industry,

a.erfasst,

a.seriell,

a.liefertermin,

a.bez,

RTRIM(a.Merchandise) AS Merchandise,

a.nr\_adress\_k,

a.cty,

a.department,

a.preis\_ges\_dm,

a.preis\_ges\_wr,

b.DB2,

a.bu\_id,

a.currency,

a.Backlogrelevant,

a.office\_reference

FROM dbo.vTMP\_VENTAS\_OrderIntake\_Comm\_02\_BU AS a

INNER JOIN dbo.vTMP\_VENTAS\_OrderIntake\_Comm\_03\_DB2pos AS b ON a.nr\_kontrakt = b.nr\_kontrakt

AND a.nr\_position = b.nr\_position

WHERE(a.erfasst >= '2008-01-01')

GO

-- Because Performamce Problem, we create two temp table (TMPLoadVentas\_Comm\_OrderIntake,TMPLoadVentas\_Comm\_OrderIntakeDB2\_Post) to store data temporarily from vTMPSourceLoadOderIntake and vTMPSourceLoadOderIntakeDB2\_Post

IF OBJECT\_ID(N'dbo.TMPLoadVentas\_Comm\_OrderIntake', N'U') IS NOT NULL

DROP TABLE dbo.TMPLoadVentas\_Comm\_OrderIntake

CREATE TABLE [dbo].[TMPLoadVentas\_Comm\_OrderIntake](

nr\_position [int] NOT NULL,

nr\_kontrakt [nvarchar](11) NULL,

abgerechnet\_dat [datetime2](7) null,

nr\_liefnt [nvarchar](8) NULL,

nr\_kunde [nvarchar](8) NULL,

industry [nvarchar](4) NULL,

erfasst [datetime2](7) NULL,

seriell int null,

liefertermin [datetime2](7) NULL,

bez [nvarchar](255) NULL,

Merchandise [nvarchar](1) NULL,

nr\_adress\_k [nvarchar](4) null,

cty [nvarchar](2) NULL,

department [nvarchar](10) NULL,

preis\_ges\_dm [money] NULL,

preis\_ges\_wr [money] NULL,

DB2 [money] NULL,

bu\_id [nvarchar](3) NULL,

currency [nvarchar](3) NULL,

Backlogrelevant [varchar](1) NULL,

office\_reference [nvarchar](30) NULL

) ON [PRIMARY]

IF OBJECT\_ID(N'dbo.TMPLoadVentas\_Comm\_OrderIntakeDB2\_Post', N'U') IS NOT NULL

DROP TABLE dbo.TMPLoadVentas\_Comm\_OrderIntakeDB2\_Post

CREATE TABLE dbo.TMPLoadVentas\_Comm\_OrderIntakeDB2\_Post

( nr\_kontrakt [nvarchar](11) NULL,

nr\_position [int] NOT NULL,

[Calculation\_Status] [bit] NULL,

[Precal\_IntakeDB2] [money] NULL,

[Precal\_IntakeDB3] [money] NULL,

[Postcal\_IntakeDB2] [money] NULL,

[Postcal\_IntakeDB3] [money] NULL,

)ON [PRIMARY]

INSERT INTO dbo.TMPLoadVentas\_Comm\_OrderIntake

SELECT \*

FROM vTMPSourceLoadOderIntake

INSERT INTO dbo.TMPLoadVentas\_Comm\_OrderIntakeDB2\_Post

SELECT \* from vTMPSourceLoadOderIntakeDB2\_Post

GO

-- Query load data

--Temp is cte table to contain all value pre value , post value , pre value DB2, Post Value DB2, Pre value DB3, Post Value DB3

WITH temp(nr\_position,

nr\_kontrakt,

abgerechnet\_dat,

nr\_liefnt,

nr\_kunde,

industry,

erfasst,

seriell,

liefertermin,

bez,

Merchandise,

nr\_adress\_k,

cty,

department,

preis\_ges\_dm,

preis\_ges\_wr,

DB2,

bu\_id,

currency,

Backlogrelevant,

office\_reference,

Precal\_IntakeDB2,

Precal\_IntakeDB2\_Currency,

Precal\_IntakeValue\_Currency,

Precal\_IntakeValue,

Precal\_IntakeDB3,

Postcal\_IntakeDB2,

Postcal\_IntakeDB2\_currency,

Postcal\_IntakeValue\_Currency,

Postcal\_IntakeValue,

Postcal\_IntakeDB3,

Calculation\_Status)

AS

(SELECT FAKT\_OrderIntake.nr\_position,

FAKT\_OrderIntake.nr\_kontrakt,

FAKT\_OrderIntake.abgerechnet\_dat,

FAKT\_OrderIntake.nr\_liefnt,

FAKT\_OrderIntake.nr\_kunde,

FAKT\_OrderIntake.industry,

FAKT\_OrderIntake.erfasst,

FAKT\_OrderIntake.seriell,

FAKT\_OrderIntake.liefertermin,

FAKT\_OrderIntake.bez,

FAKT\_OrderIntake.Merchandise,

FAKT\_OrderIntake.nr\_adress\_k,

FAKT\_OrderIntake.cty,

FAKT\_OrderIntake.department,

FAKT\_OrderIntake.preis\_ges\_dm,

FAKT\_OrderIntake.preis\_ges\_wr,

FAKT\_OrderIntake.DB2,

FAKT\_OrderIntake.bu\_id,

FAKT\_OrderIntake.currency,

FAKT\_OrderIntake.Backlogrelevant,

FAKT\_OrderIntake.office\_reference,

vTMP\_VENTAS\_OrderComplete\_Comm\_DB2pos.Precal\_IntakeDB2,

'0' AS Precal\_IntakeDB2\_Currency,

dbo.vTMP\_VENTAS\_OrderComplete\_Comm\_Value.PreCal\_Value\_Currency AS Precal\_IntakeValue\_Currency,

dbo.vTMP\_VENTAS\_OrderComplete\_Comm\_Value.PreCal\_Value AS Precal\_IntakeValue,

vTMP\_VENTAS\_OrderComplete\_Comm\_DB2pos.Precal\_IntakeDB3,

vTMP\_VENTAS\_OrderComplete\_Comm\_DB2pos.Postcal\_IntakeDB2,

'0' AS Postcal\_IntakeDB2\_currency,

dbo.vTMP\_VENTAS\_OrderComplete\_Comm\_Value.PostCal\_Value\_Currency AS Postcal\_IntakeValue\_Currency,

dbo.vTMP\_VENTAS\_OrderComplete\_Comm\_Value.PostCal\_Value AS Postcal\_IntakeValue,

vTMP\_VENTAS\_OrderComplete\_Comm\_DB2pos.Postcal\_IntakeDB3,

vTMP\_VENTAS\_OrderComplete\_Comm\_DB2pos.Calculation\_Status

FROM dbo.TMPLoadVentas\_Comm\_OrderIntake AS FAKT\_OrderIntake

LEFT JOIN dbo.vTMP\_VENTAS\_OrderComplete\_Comm\_Value ON dbo.vTMP\_VENTAS\_OrderComplete\_Comm\_Value.nr\_position = FAKT\_OrderIntake.nr\_position

AND dbo.vTMP\_VENTAS\_OrderComplete\_Comm\_Value.nr\_kontrakt = FAKT\_OrderIntake.nr\_kontrakt

LEFT JOIN dbo.TMPLoadVentas\_Comm\_OrderIntakeDB2\_Post vTMP\_VENTAS\_OrderComplete\_Comm\_DB2pos ON vTMP\_VENTAS\_OrderComplete\_Comm\_DB2pos.nr\_kontrakt = FAKT\_OrderIntake.nr\_kontrakt

AND vTMP\_VENTAS\_OrderComplete\_Comm\_DB2pos.nr\_position = FAKT\_OrderIntake.nr\_position)

--- this is the process to solve null value from temp data

SELECT a.nr\_position,

a.nr\_kontrakt,

a.abgerechnet\_dat,

a.nr\_liefnt,

a.nr\_kunde,

a.industry,

a.erfasst,

a.seriell,

a.liefertermin,

a.bez,

RTRIM(a.Merchandise) AS Merchandise,

a.nr\_adress\_k,

a.cty,

a.department,

a.preis\_ges\_dm,

a.preis\_ges\_wr,

a.DB2,

a.bu\_id,

a.currency,

a.Backlogrelevant,

a.office\_reference,

CASE

WHEN ISNULL(a.Precal\_IntakeValue, 0) = 0

THEN a.preis\_ges\_dm

ELSE ISNULL(a.Precal\_IntakeValue, 0)

END AS Precal\_IntakeValue,

CASE

WHEN ISNULL(a.Precal\_IntakeDB2, 0) = 0

THEN a.DB2

ELSE ISNULL(a.Precal\_IntakeDB2, 0)

END AS Precal\_IntakeDB2,

ISNULL(a.Precal\_IntakeDB2\_Currency, 0) AS Precal\_IntakeDB2\_Currency,

CASE

WHEN ISNULL(a.Precal\_IntakeValue\_Currency, 0) = 0

THEN a.preis\_ges\_wr

ELSE ISNULL(a.Precal\_IntakeValue\_Currency, 0)

END AS Precal\_IntakeValue\_Currency,

ISNULL(a.Postcal\_IntakeValue, 0) AS Postcal\_IntakeValue,

ISNULL(a.Postcal\_IntakeValue\_Currency, 0) AS Postcal\_IntakeValue\_Currency,

ISNULL(a.Postcal\_IntakeDB2, 0) AS Postcal\_IntakeDB2,

ISNULL(a.Postcal\_IntakeDB2\_currency, 0) AS Postcal\_IntakeDB2\_currency,

ISNULL(a.Precal\_IntakeDB3, 0) AS Precal\_IntakeDB3,

ISNULL(a.Postcal\_IntakeDB3, 0) AS Postcal\_IntakeDB3,

CASE

WHEN a.[Calculation\_Status] IS NULL

THEN 0

ELSE a.[Calculation\_Status]

END AS Calculation\_Status

FROM temp AS a

**New Mapping :**

Precal\_IntakeValue 🡪Precal\_IntakeValue,

Precal\_IntakeDB2🡪Precal\_IntakeDB2,

Precal\_IntakeDB2\_Currency 🡪Precal\_IntakeDB2\_Currency,

Precal\_IntakeValue\_Currency 🡪Precal\_IntakeValue\_Currency,

Postcal\_IntakeValue 🡪Postcal\_IntakeValue,

Postcal\_IntakeValue\_Currency 🡪Postcal\_IntakeValue\_Currency,

Postcal\_IntakeDB2🡪Postcal\_IntakeDB2,

Postcal\_IntakeDB2\_currency 🡪Postcal\_IntakeDB2\_currency,

Precal\_IntakeDB3🡪Precal\_IntakeDB3,

Postcal\_IntakeDB3🡪Postcal\_IntakeDB3,

[Calculation\_Status]🡪[Calculation\_Status]

### 6.1.3. Foxpro\_OrderIntake

**Original Source Query Load**

SELECT

ord\_date,

ord\_no,

division,

industry\_code,

bu\_id,

cty,

merchandise\_id,

descriptionofgoods,

Salesman,

sp\_centre,

CASE WHEN direct\_ord = 1 THEN 'C' ELSE 'T' END AS ordertype,

service,

contr\_no,

shp\_date,

contr\_date,

customer,

province,

pro\_centre,

maker AS supplyer,

commodity,

OrderValue\_currency\_type,

OrderValue\_in\_local\_currency,

OrderValue,

DB2\_currency\_type,

DB2\_in\_local\_currency,

DB2,

1 AS orderpos,

shipped,

'J' AS backlogrelevant,

ord\_no AS office\_reference

FROM

dbo.vTMP\_Foxpro\_OrderIntake

WHERE

ord\_date >= '2008-01-01'

and not (OrderValue = 1 and DB2 = 0) -- no canceled orders

**Mapping:**

OrderValue 🡪 orderintake\_value

OrderValue\_in\_local\_currency 🡪 orderintake\_value\_currency

DB2 🡪 orderintake\_db2

DB2\_in\_local\_currency🡪 orderintake\_db2\_currency

**Current Logic data :get all value Pre of all orderno**

**Change Logic data :**

/\*FOXPRO\_Order\_Intake\*/

-- FOXPRO\_Order\_Intake have only Pre Value of orders from this source

SELECT ord\_date,

ord\_no,

division,

industry\_code,

bu\_id,

cty,

merchandise\_id,

descriptionofgoods,

Salesman,

sp\_centre,

CASE

WHEN direct\_ord = 1

THEN 'C'

ELSE 'T'

END AS ordertype,

service,

contr\_no,

shp\_date,

contr\_date,

customer,

province,

pro\_centre,

maker AS supplyer,

commodity,

OrderValue\_currency\_type,

OrderValue\_in\_local\_currency,

OrderValue,

DB2\_currency\_type,

DB2\_in\_local\_currency,

DB2,

1 AS orderpos,

shipped,

'J' AS backlogrelevant,

ord\_no AS office\_reference,

OrderValue AS Precal\_IntakeValue,

DB2 AS Precal\_IntakeDB2,

DB2\_in\_local\_currency AS Precal\_IntakeDB2\_Currency,

OrderValue\_in\_local\_currency AS Precal\_IntakeValue\_Currency,

'0' AS Postcal\_IntakeValue,

'0' AS Postcal\_IntakeValue\_Currency,

'0' AS Postcal\_IntakeDB2,

'0' AS Postcal\_IntakeDB2\_currency,

'0' AS Precal\_IntakeDB3,

'0' AS Postcal\_IntakeDB3,

'0' AS [Calculation\_Status]

FROM dbo.vTMP\_Foxpro\_OrderIntake

WHERE ord\_date >= '2008-01-01'

AND NOT(OrderValue = 1

AND DB2 = 0); -- no canceled orders

**New Mapping :**

Precal\_IntakeValue 🡪Precal\_IntakeValue,

Precal\_IntakeDB2🡪Precal\_IntakeDB2,

Precal\_IntakeDB2\_Currency 🡪Precal\_IntakeDB2\_Currency,

Precal\_IntakeValue\_Currency 🡪Precal\_IntakeValue\_Currency,

Postcal\_IntakeValue 🡪Postcal\_IntakeValue,

Postcal\_IntakeValue\_Currency 🡪Postcal\_IntakeValue\_Currency,

Postcal\_IntakeDB2🡪Postcal\_IntakeDB2,

Postcal\_IntakeDB2\_currency 🡪Postcal\_IntakeDB2\_currency,

Precal\_IntakeDB3🡪Precal\_IntakeDB3,

Postcal\_IntakeDB3🡪Postcal\_IntakeDB3,

[Calculation\_Status]🡪[Calculation\_Status]

### 6.1.4. Local Orders\_OrderIntake

**Original Source Query Load**

SELECT ID,

ProcCenterOrderNo,

Order\_Date,

Shipment\_Date,

bu\_id,

OfficeLocation,

AbtAusBuchKz,

LandAusBuchKz,

Industriebereich,

Vorgang,

KdNr,

Kurs,

LieferantNr,

Warenbez,

DB2,

EK\_Preis,

VK\_Wrg,

Wert\_Wrg,

MakerComm,

Customs,

SubAgent,

Freight,

Banking,

TravellingExp,

Commissioning,

Service,

Contingencies,

bu\_type,

LieferterminDat,

Zahlungsbed,

[User],

ArtikelAusGA,

source,

orderpos,

vorgang AS office\_reference,

Backlogrelevant

FROM dbo.vTMP\_HKGOrders\_DB2

WHERE ISNUMERIC(bu\_id) = 1;

**Mapping:**

Wert\_Wrg 🡪 orderintake\_value

DB2 🡪 orderintake\_db2

**Current Logic data :get all value Pre of all orderno**

**Change Logic data :**

/\*Local Orders\_OrderIntake\*/

-- Because Local Orders only have Pre value so that Local Orders don't have post value

SELECT ID,

ProcCenterOrderNo,

Order\_Date,

Shipment\_Date,

bu\_id,

OfficeLocation,

AbtAusBuchKz,

LandAusBuchKz,

Industriebereich,

Vorgang,

KdNr,

Kurs,

LieferantNr,

Warenbez,

DB2,

EK\_Preis,

VK\_Wrg,

Wert\_Wrg,

MakerComm,

Customs,

SubAgent,

Freight,

Banking,

TravellingExp,

Commissioning,

Service,

Contingencies,

bu\_type,

LieferterminDat,

Zahlungsbed,

[User],

ArtikelAusGA,

source,

orderpos,

vorgang AS office\_reference,

Backlogrelevant,

Wert\_Wrg AS Precal\_IntakeValue,

DB2 AS Precal\_IntakeDB2,

'0' AS Precal\_IntakeDB2\_Currency,

'0' AS Precal\_IntakeValue\_Currency,

'0' AS Postcal\_IntakeValue,

'0' AS Postcal\_IntakeValue\_Currency,

'0' AS Postcal\_IntakeDB2,

'0' AS Postcal\_IntakeDB2\_currency,

'0' AS Precal\_IntakeDB3,

'0' AS Postcal\_IntakeDB3,

'0' AS [Calculation\_Status]

FROM dbo.vTMP\_HKGOrders\_DB2

WHERE ISNUMERIC(bu\_id) = 1;

**New Mapping :**

Precal\_IntakeValue 🡪Precal\_IntakeValue,

Precal\_IntakeDB2🡪Precal\_IntakeDB2,

Precal\_IntakeDB2\_Currency 🡪Precal\_IntakeDB2\_Currency,

Precal\_IntakeValue\_Currency 🡪Precal\_IntakeValue\_Currency,

Postcal\_IntakeValue 🡪Postcal\_IntakeValue,

Postcal\_IntakeValue\_Currency 🡪Postcal\_IntakeValue\_Currency,

Postcal\_IntakeDB2🡪Postcal\_IntakeDB2,

Postcal\_IntakeDB2\_currency 🡪Postcal\_IntakeDB2\_currency,

Precal\_IntakeDB3🡪Precal\_IntakeDB3,

Postcal\_IntakeDB3🡪Postcal\_IntakeDB3,

[Calculation\_Status]🡪[Calculation\_Status]

### 6.1.5. LOD Order Intake

**Original Source Query Load**

select \* from vTMP\_LOD\_OrderIntake

VIEW [dbo].[vTMP\_LOD\_OrderIntake]

AS

SELECT o.Industry,

o.Merchandise,

o.EstDeliveryDate,

o.Currency,

o.OrderIntakeDate,

o.GoodDescription,

o.OrderVal\_EUR,

o.DB2Val\_EUR,

o.OrderVal,

o.DB2Val,

o.LocalRef,

c.CustomerID,

s.SupplierID,

o.GroupNo AS OrderNo\_New,

CAST('LOD' AS NVARCHAR(5)) AS source,

tb.bu\_id,

CASE

WHEN vl.source = 'HKG'

THEN 'J'

ELSE CASE

WHEN bs.ShippedDate IS NOT NULL

THEN 'N'

ELSE 'J'

END

END AS backlogrelevant,

'1' AS OrderPos,

bs.ShippedDate,

LEFT(o.BusinessType, 1) AS OrderType,

CAST(pc.code AS NVARCHAR(3)) AS ProcurementCenter,

ors.LabourDB3,

ors.LabourDB3Post

FROM LZ\_LOD\_OrderReg o

LEFT JOIN LZ\_LOD\_Customer c ON o.Customer = c.CustomerName

LEFT JOIN LZ\_LOD\_Supplier s ON o.Supplier = s.SupplierName

JOIN tmp\_bu tb ON o.BusinessUnit = tb.bu\_desc

JOIN

(

SELECT OrderNo,

'HKG' AS source

FROM dbo.vLOD\_HKGOrderNo

UNION ALL

SELECT OrderNo,

'HKG'

FROM dbo.vLOD\_FOXPROOrderNo

UNION ALL

SELECT OrderNo,

'nonHKG'

FROM dbo.vLOD\_NonHKGOrderNo

) vl ON o.GroupNo = vl.OrderNo

LEFT JOIN LZ\_LOD\_OrderBillingShipping bs ON o.OrderNo = bs.OrderNo

LEFT JOIN ADM\_ProcurementCenter pc ON pc.description = o.ProcCenter

LEFT JOIN LZ\_LOD\_OrderRegService ors ON o.OrderNo = ors.OrderNo

AND o.Merchandise = 'S'

WHERE ISNULL(ors.Chareable, 1) <> 0; -- filter out non-chargeable LOD service orders

**Mapping:**

OrderVal\_EUR 🡪 orderintake\_value

OrderVal 🡪 orderintake\_value\_currency

DB2Val\_EUR🡪 orderintake\_db2

DB2Val 🡪 orderintake\_db2\_currency

LabourDB3🡪orderintake\_db3

**Current Logic data :get all value Pre of all orderno**

**Change Logic data :**

VIEW [dbo].[vTMP\_LOD\_OrderIntake]

AS

SELECT o.Industry,

o.Merchandise,

o.EstDeliveryDate,

o.Currency,

o.OrderIntakeDate,

o.GoodDescription,

o.OrderVal\_EUR,

o.DB2Val\_EUR,

o.OrderVal,

o.DB2Val,

o.LocalRef,

c.CustomerID,

s.SupplierID,

o.GroupNo AS OrderNo\_New,

CAST('LOD' AS NVARCHAR(5)) AS source,

tb.bu\_id,

CASE

WHEN vl.source = 'HKG'

THEN 'J'

ELSE CASE

WHEN bs.ShippedDate IS NOT NULL

THEN 'N'

ELSE 'J'

END

END AS backlogrelevant,

'1' AS OrderPos,

bs.ShippedDate,

LEFT(o.BusinessType, 1) AS OrderType,

CAST(pc.code AS NVARCHAR(3)) AS ProcurementCenter,

ors.LabourDB3,

ors.LabourDB3Post,

o.OrderVal\_EUR AS Precal\_IntakeValue,

o.DB2Val\_EUR AS Precal\_IntakeDB2,

o.OrderVal AS Precal\_IntakeValue\_Currency,

o.DB2Val AS Precal\_IntakeDB2\_Currency,

ors.LabourDB3 AS Precal\_IntakeDB3,

-- get post Value case when Post value get from LOD or Post value get from Solomon

CASE

WHEN bs.ShippedDate IS NOT NULL

AND bs.FinalizedDate IS NOT NULL

AND YEAR(o.OrderIntakeDate) < 2013

THEN ISNULL(bs.OrderVal\_EUR, 0)

WHEN ss.active = 0

AND YEAR(o.OrderIntakeDate) >= 2013

AND o.OrderIntakeDate >= '2008-01-01'

THEN ISNULL(sc.OrderCompleteValue, 0)

ELSE 0

END AS Postcal\_IntakeValue,

CASE

WHEN bs.ShippedDate IS NOT NULL

AND bs.FinalizedDate IS NOT NULL

AND YEAR(o.OrderIntakeDate) < 2013

THEN ISNULL(bs.OrderVal, 0)

WHEN ss.active = 0

AND YEAR(o.OrderIntakeDate) >= 2013

AND o.OrderIntakeDate >= '2008-01-01'

THEN ISNULL(o.OrderVal, 0)

ELSE 0

END AS Postcal\_IntakeValue\_Currency,

CASE

WHEN bs.ShippedDate IS NOT NULL

AND bs.FinalizedDate IS NOT NULL

AND YEAR(o.OrderIntakeDate) < 2013

THEN ISNULL(bs.DB2Amt\_EUR, 0)

WHEN ss.active = 0

AND YEAR(o.OrderIntakeDate) >= 2013

AND o.OrderIntakeDate >= '2008-01-01'

THEN ISNULL(sd.OrderCompleteDB2, 0)

ELSE 0

END AS Postcal\_IntakeDB2,

CASE

WHEN bs.ShippedDate IS NOT NULL

AND bs.FinalizedDate IS NOT NULL

AND YEAR(o.OrderIntakeDate) < 2013

THEN ISNULL(bs.DB2Amt, 0)

WHEN ss.active = 0

AND YEAR(o.OrderIntakeDate) >= 2013

AND o.OrderIntakeDate >= '2008-01-01'

THEN 0

ELSE 0

END AS Postcal\_IntakeDB2\_currency,

CASE

WHEN bs.ShippedDate IS NOT NULL

AND bs.FinalizedDate IS NOT NULL

AND YEAR(o.OrderIntakeDate) < 2013

THEN ISNULL(ors.LabourDB3Post, 0)

WHEN ss.active = 0

AND YEAR(o.OrderIntakeDate) >= 2013

AND o.OrderIntakeDate >= '2008-01-01'

THEN 0

ELSE 0

END AS Postcal\_IntakeDB3,

CASE

WHEN bs.ShippedDate IS NOT NULL

AND bs.FinalizedDate IS NOT NULL

AND YEAR(o.OrderIntakeDate) < 2013

THEN 1

WHEN ss.active = 0

AND YEAR(o.OrderIntakeDate) >= 2013

AND o.OrderIntakeDate >= '2008-01-01'

THEN 1

ELSE 0

END AS Calculation\_Status

FROM LZ\_LOD\_OrderReg o

LEFT JOIN LZ\_LOD\_Customer c ON o.Customer = c.CustomerName

LEFT JOIN LZ\_LOD\_Supplier s ON o.Supplier = s.SupplierName

JOIN tmp\_bu tb ON o.BusinessUnit = tb.bu\_desc

JOIN

(

SELECT OrderNo,

'HKG' AS source

FROM dbo.vLOD\_HKGOrderNo

UNION ALL

SELECT OrderNo,

'HKG'

FROM dbo.vLOD\_FOXPROOrderNo

UNION ALL

SELECT OrderNo,

'nonHKG'

FROM dbo.vLOD\_NonHKGOrderNo

) vl ON o.GroupNo = vl.OrderNo

LEFT JOIN LZ\_LOD\_OrderBillingShipping bs ON o.OrderNo = bs.OrderNo

LEFT JOIN ADM\_ProcurementCenter pc ON pc.description = o.ProcCenter

LEFT JOIN LZ\_LOD\_OrderRegService ors ON o.OrderNo = ors.OrderNo

AND o.Merchandise = 'S'

LEFT JOIN dbo.vTMP\_Solomon\_03\_OrderComplete\_Date\_BusinessType\_new sb ON o.GroupNo = sb.ord\_no

LEFT JOIN dbo.vTMP\_Solomon\_02\_OrderComplete\_value\_new sc ON o.GroupNo = sc.ord\_no

LEFT JOIN

-- get Ordercomplete DB2 group by OrderNo. because vTMP\_Solomon\_02\_OrderComplete\_DB2\_New have DB2 value group by Orderno and AccountNo

(

SELECT [ord\_no],

SUM([OrderCompleteDB2]) AS [OrderCompleteDB2]

FROM [dbo].[vTMP\_Solomon\_02\_OrderComplete\_DB2\_New]

GROUP BY [ord\_no]

) AS sd ON o.GroupNo = sd.ord\_no

LEFT JOIN dbo.TMP\_Solomon\_SubAcct ss ON o.GroupNo = ss.orderno

WHERE ISNULL(ors.Chareable, 1) <> 0; -- filter out non-chargeable LOD service orders

**New Mapping :**

Precal\_IntakeValue 🡪Precal\_IntakeValue,

Precal\_IntakeDB2🡪Precal\_IntakeDB2,

Precal\_IntakeDB2\_Currency 🡪Precal\_IntakeDB2\_Currency,

Precal\_IntakeValue\_Currency 🡪Precal\_IntakeValue\_Currency,

Postcal\_IntakeValue 🡪Postcal\_IntakeValue,

Postcal\_IntakeValue\_Currency 🡪Postcal\_IntakeValue\_Currency,

Postcal\_IntakeDB2🡪Postcal\_IntakeDB2,

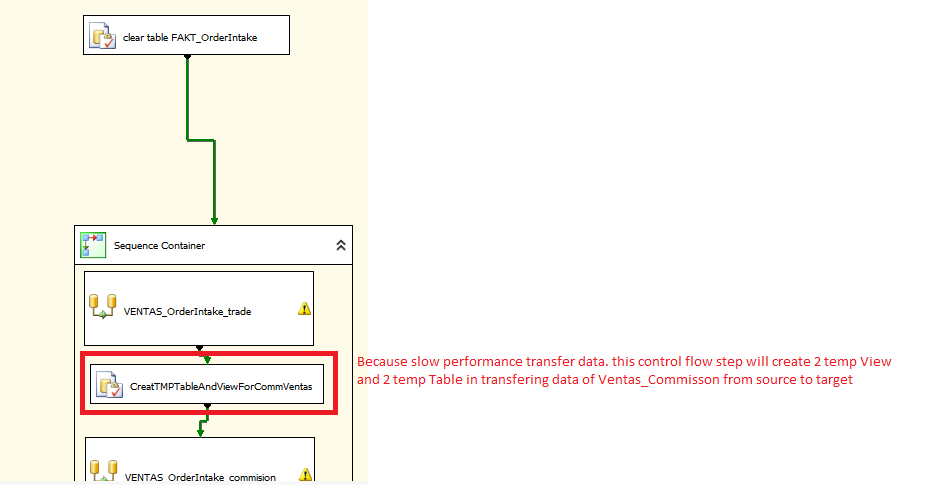
Postcal\_IntakeDB2\_currency 🡪Postcal\_IntakeDB2\_currency,

Precal\_IntakeDB3🡪Precal\_IntakeDB3,

Postcal\_IntakeDB3🡪Postcal\_IntakeDB3,

[Calculation\_Status]🡪[Calculation\_Status]

### 6.1.6. Change SSIS package



* 1. Source query loading change

**Old source load** **new source load**

1. Test Process
   1. Fakt\_OrderComplete and Fakt\_OrderCompleteDB2

Local test environment

**Step 1 :**

Clone all structure staging database to Staging\_Model Database by using SQL compare

**Step 2:**

Deploy changes structure to Staging\_Model

**Step 3:**

Restore database staging as new database with new name StagingTest

**Step 4:** Deploy new structure from Staging\_Model to StagingTest

**Step 5:** Run SSIS package Fact\_OrderComplete with connecting to StagingTest

**Step 6** :Test Case compare data of StagingTest with Staging



* 1. Fakt\_OrderIntake

Local test environment

**Step 1 :**

Clone all structure staging database to Staging\_Model Database by using SQL compare

**Step 2:**

Deploy changes structure to Staging\_Model

**Step 3:**

Restore database staging as new database with new name StagingTest

**Step 4:** Deploy new structure from Staging\_Model to StagingTest

**Step 5:** Run SSIS package Fact\_OrderComplete with connecting to StagingTest

**Step 6 :**Test Case compare data of StagingTest with Staging



* 1. Board