**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**

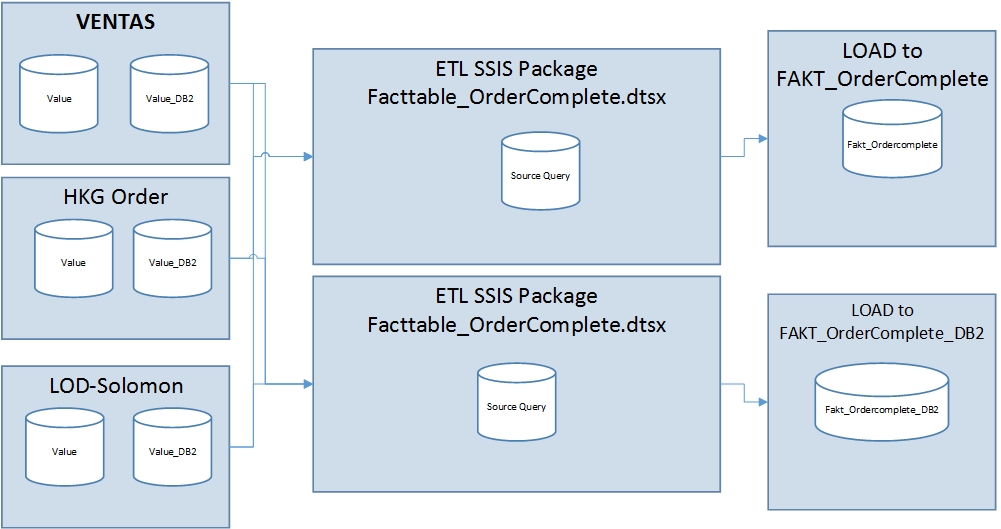
* Development change only impacts on logical data of two tables that are FAKT\_OrderComplete table and FAKT\_OrderComplete\_DB2 table. Because the distinction data in (ordercomplete\_value and ordercomplete\_value\_DB2) is not yet clear and don’t know those values belong to Pre-calculation or Post-Calculation
* Board layer : will be had new two version

The first version : with Pre-calculated values ONLY, so always pre-calc., even if post-calc. is available

The Second Version : take post-calc

Between two version will be distinguished by calculation\_status

1. **Evaluation of Current System**
   1. Board
   2. DWH
      1. Context Diagram LoadToFact



* + 1. Structure Table and View
* 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**
   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

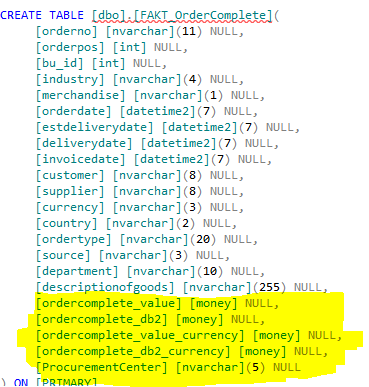
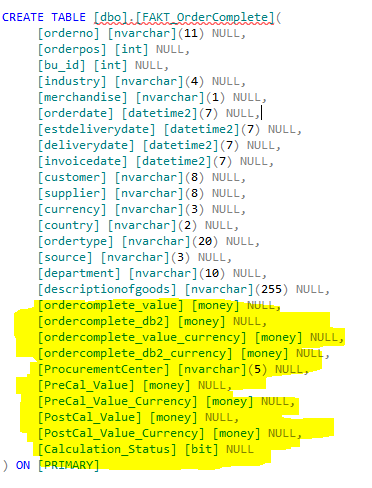
PostCal\_Value\_DB3 🡪 PostCal\_Value\_DB3

Calculation\_Status 🡪 Calculation\_Status

1. Implementation
   1. DWH
      1. Staging Structure Database

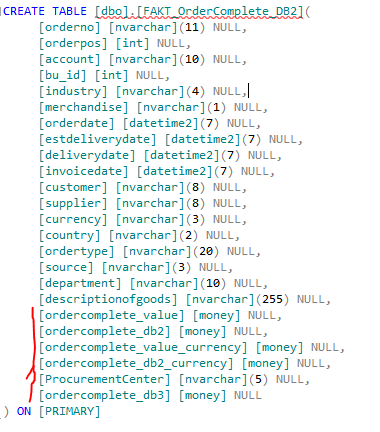
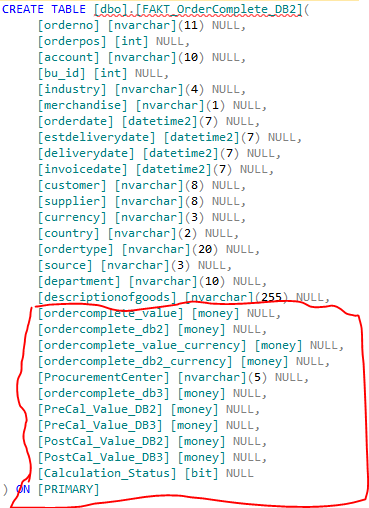
* Change Structure FAKT\_OrderComplete table by adding new five columns

Old structure New structure



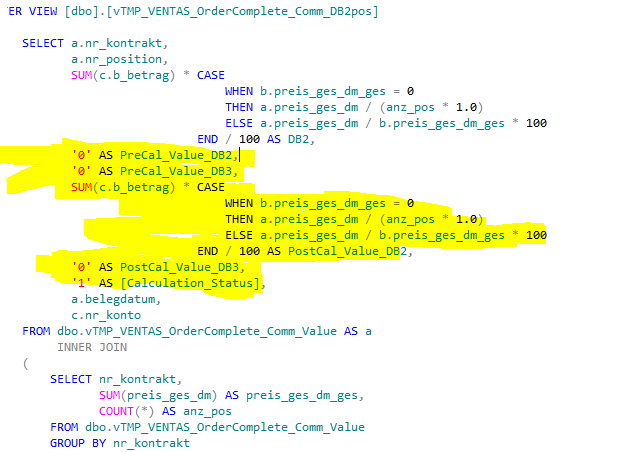
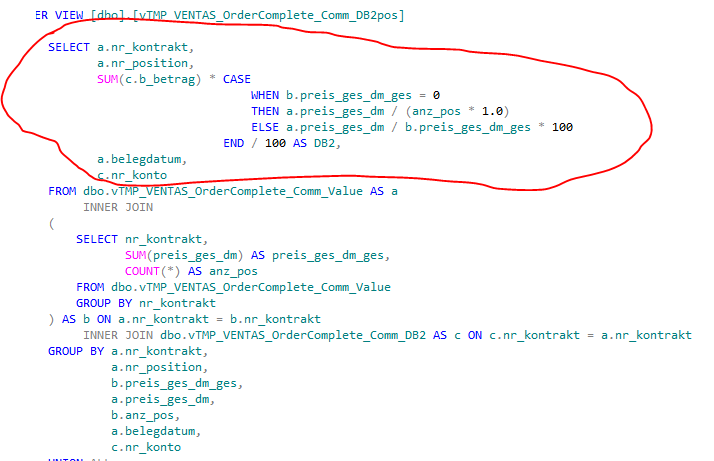
* Change structure FAKT\_OrderComplete\_DB2 table by adding new five columns

Old Structure New Structure



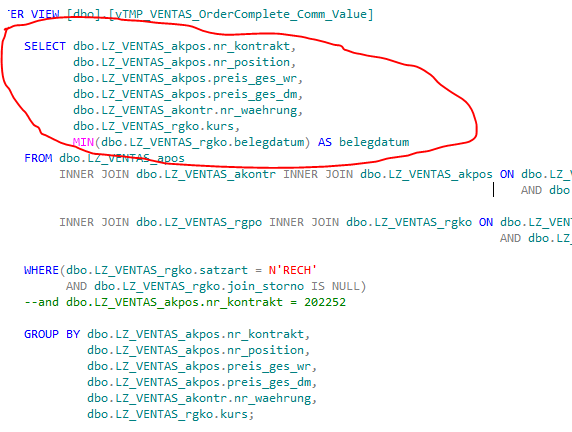
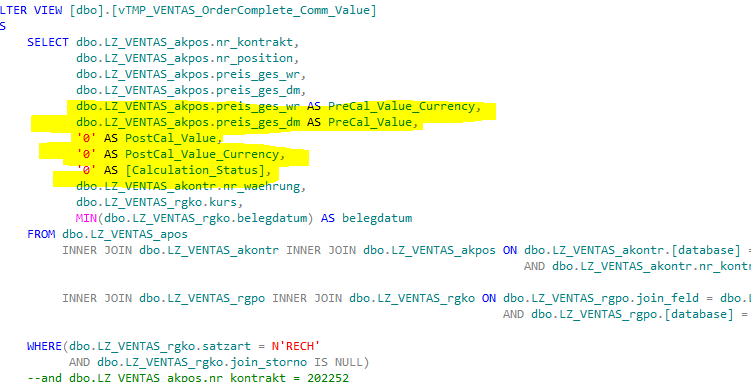
* Change Structure view [vTMP\_VENTAS\_OrderComplete\_Comm\_DB2pos]

Old Structure New Structure



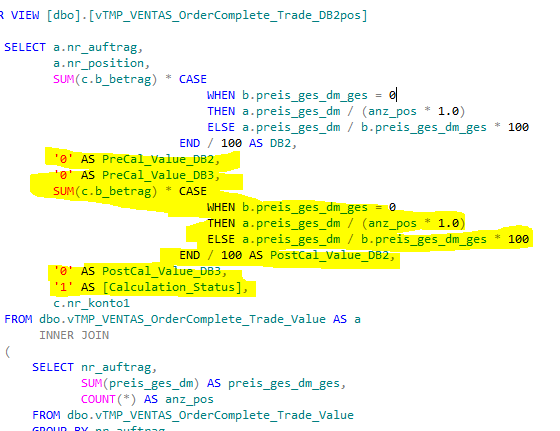
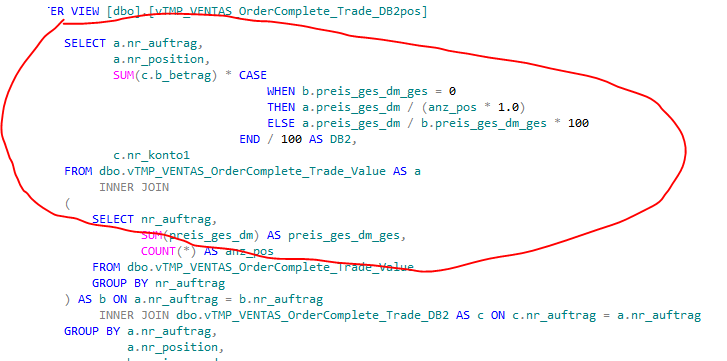
* Change structure view [vTMP\_VENTAS\_OrderComplete\_Comm\_Value]

Old Structure New Structure



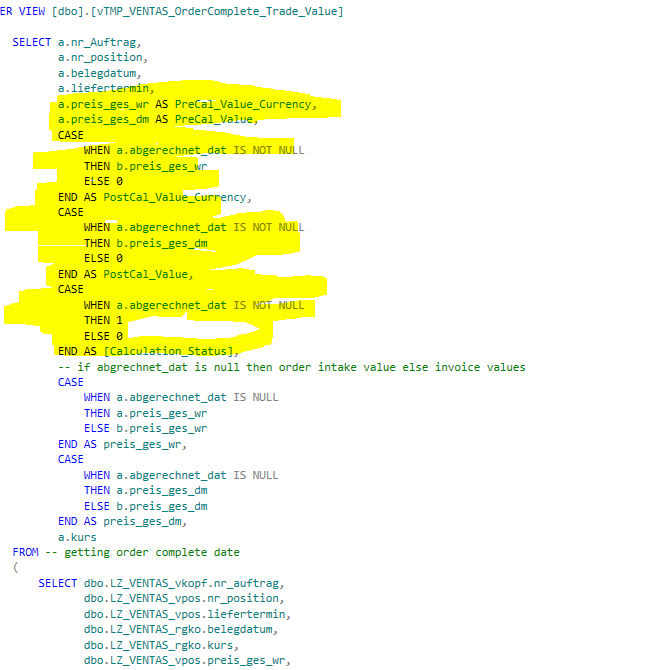
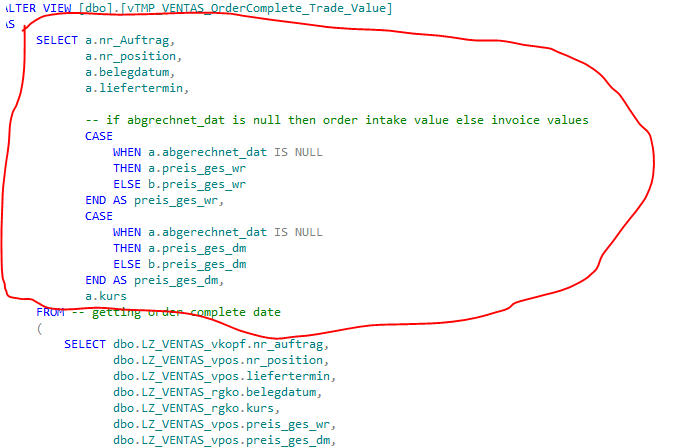
* Change structure view [vTMP\_VENTAS\_OrderComplete\_Trade\_DB2pos]

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. Test Process
   1. DWH

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

1. Deployment