BAB III

PELAKSANAAN KERJA

3.1 Kedudukan dan Koordinasi

Pemahaman terhadap kedudukan dan mekanisme koordinasi ini penting untuk menggambarkan bagaimana tugas yang berkaitan dengan proyek "Migrasi Data SAP Business Warehousing ke ABAP Core Data Services pada SAP S/4HANA: Studi Kasus PT X di PT W" dijalankan secara sistematis dan profesional.

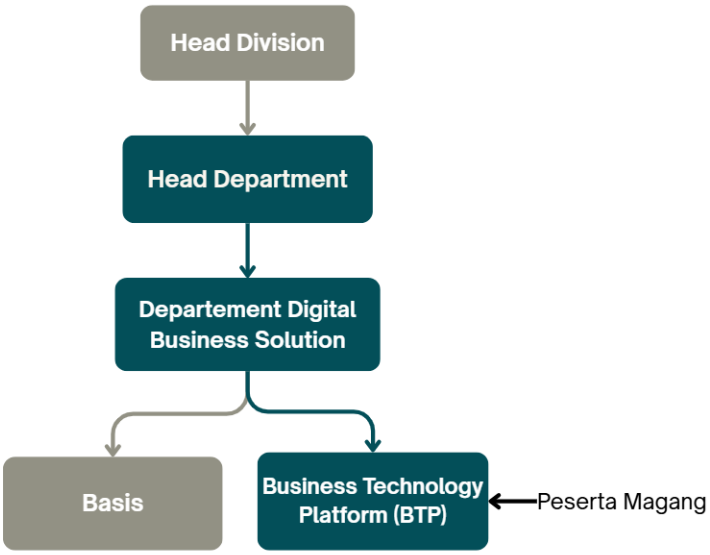
3.1.1 Kedudukan

Secara formal, peserta magang (intern) di PT W ditempatkan pada Departemen Digital Business Solution yang berada di bawah Head Department, dan selanjutnya berada di bawah Head Division yang mengelola keseluruhan aktivitas divisi terkait solusi digital berbasis SAP.

Dengan penempatan sebagai Consultant Intern, peran yang serupa dengan konsultannya W tingkat junior dijalankan. Tugas tidak hanya berfokus pada coding, tetapi juga meliputi analisis kebutuhan, review logika pelaporan, diskusi desain data model dengan mentor, serta penyusunan dokumentasi teknis. Standar dokumentasi dan coding yang berlaku di W harus diikuti, sebagaimana konsultan-konsultan lain dalam proyek.

Di dalam Departemen Digital Business Solution, Tim Business Technology Platform (BTP) merupakan tim yang menangani pengembangan solusi berbasis SAP Business Technology Platform, integrasi data, serta pengembangan artefak teknis seperti ABAP Core Data Services (CDS) dan komponen pendukungnya. Tim BTP ini berada sejajar

dengan Tim Basis yang mengelola aspek infrastruktur dan administrasi teknis sistem SAP. Secara garis besar, intern dalam struktur organisasi divisi yang relevan dapat digambarkan sebagai berikut:

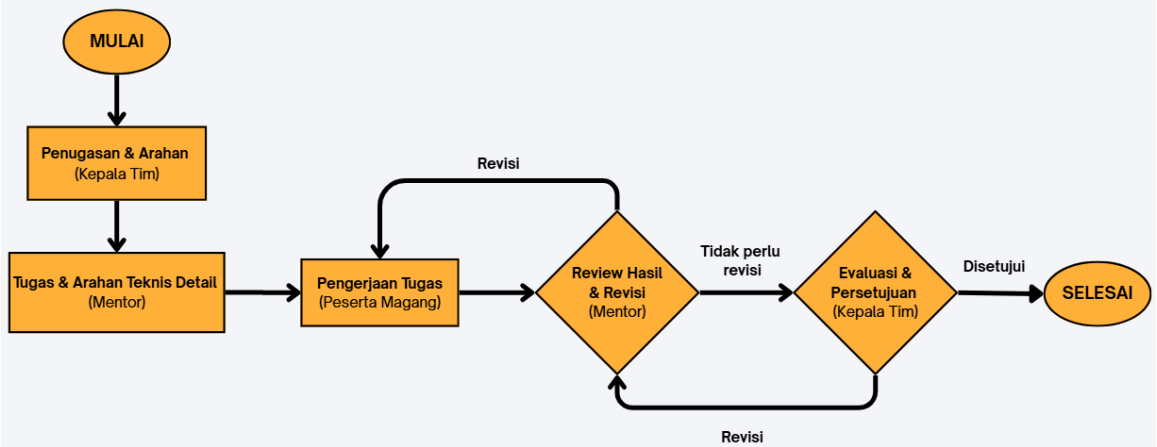


Gambar 3.1 Kedudukan Peserta Magang dalam Struktur Organisasi

Dalam struktur tersebut, peserta magang berada langsung di Tim Business Technology Platform (BTP) dan bekerja sehari-hari bersama anggota tim lain. Membantu proses analisis, pemodelan, dan pengembangan CDS view serta aktivitas terkait migrasi data dari SAP BW ke SAP S/4HANA sesuai arahan mentor dan kepala tim adalah tanggung jawab utama. Dengan kedudukan ini, keterlibatan langsung dalam siklus pengembangan solusi, mulai dari pemahaman kebutuhan hingga uji coba dan dokumentasi hasil menjadi kesempatan yang dimiliki.

3.1.2 Koordinasi

Alur koordinasi kerja selama magang dirancang agar jelas dan efisien, dengan memanfaatkan peran mentor dan kepala tim sebagai pengarah utama. Secara umum, pola koordinasi yang berlangsung adalah sebagai berikut:



Gambar 3.2 Bagan Alur Koordinasi Kerja

1) **Mulai -> Penugasan & Arahan (Kepala Tim)**

Pekerjaan dimulai ketika kepala tim memberikan penugasan dan arahan umum terkait task/proyek yang harus dikerjakan.

2) **Tugas & Arahan Teknis Detail (Mentor)**

Arahan dari kepala tim kemudian diteruskan ke mentor. Mentor menerjemahkan arahan tersebut menjadi tugas teknis yang lebih rinci.

3) **Pengerjaan Tugas**

Berdasarkan instruksi teknis dari mentor, tugas (misalnya membuat CDS, blueprint, mapping, dll) dikerjakan.

4) **Review Hasil & Revisi (Mentor)**

Setelah selesai, hasil pekerjaan diserahkan ke mentor untuk direview.

a) Jika perlu revisi, mentor memberi masukan agar pekerjaan diperbaiki, lalu kembali lagi ke tahap review mentor.
b) Jika tidak perlu revisi, hasil dinyatakan cukup dan diteruskan ke kepala tim.

5) **Evaluasi & Persetujuan (Kepala Tim)**

Kepala tim mengevaluasi hasil akhir:

a) Jika disetujui, proses dinyatakan selesai.
b) Jika perlu revisi, kepala tim memberi masukan tambahan. Revisi ini dikoordinasikan kembali melalui mentor dan peserta magang hingga hasil sesuai harapan.

3.2 Tugas yang Dilakukan

Berisi tabel hal-hal yang dilakukan selama menjalankan program.

Tabel 3.1 Detail Pekerjaan yang Dilakukan

No.	Proyek	Minggu	Keterangan
1	Pengenalan Lingkungan Kerja & Analisis Blueprint BW	Minggu 1–2 (September 2025)	Masuk ke sistem melalui SAP Logon dan menyiapkan environment pengembangan di Eclipse (ABAP Development Tools). Mempelajari blueprint/mapping BW dari PT X, memahami struktur InfoProvider, field, key figure, dan kebutuhan pelaporan yang akan dijadikan dasar perancangan CDS.
2	Perancangan CDS View Awal	Minggu 2–3 (September 2025)	Menelusuri dan menentukan asal field dari berbagai I_View/tabel standar SAP, membangun CDS basic view dan cube view awal melalui join antar I_View dan master

	Berdasarkan Blueprint		data, serta mulai menerapkan function, formula, dan filter dasar sesuai blueprint BW.
3	Penyusunan Mockup, Blueprint CDS & Penyempurnaan Struktur Join	Minggu 3–4 (September 2025) & Minggu 1–2 (Oktober 2025)	Menyusun mockup tampilan laporan dan blueprint CDS (basic–cube–consumption), kemudian mempresentasikannya kepada mentor dan kepala tim. Melakukan revisi berdasarkan masukan, merapikan join, alias, dan field, serta memastikan desain CDS selaras dengan kebutuhan bisnis PT X.
4	Redesain Struktur Mapping CDS & Penyederhanaan Arsitektur	Minggu 2–4 (Oktober 2025)	Mengidentifikasi masalah struktur awal (join/union kompleks dan data sulit ter-join), lalu bersama mentor menyusun struktur mapping CDS baru: join dilakukan sejak awal dan hanya field penting yang diteruskan. Mengimplementasikan desain baru di Eclipse dan membandingkan hasil CDS dengan laporan BW untuk memastikan konsistensi data.
5	Pengujian Performa, Konversi AMDP ke CDS & Penanganan Duplicate Data	Minggu 1–3 (November 2025)	Melakukan speed test CDS, menganalisis waktu eksekusi, dan menemukan bagian yang masih lambat. Mengonversi beberapa AMDP Table Function ke CDS/Table Function untuk meningkatkan performa, kemudian menguji ulang. Mengidentifikasi dan menangani duplicate data setelah konversi dengan memperbaiki kunci join, agregasi, serta memverifikasi kembali hasil dengan referensi BW.
6	Finalisasi & Penambahan CDS Tambahan untuk S/4HANA W	Minggu 4 (November 2025) & Minggu 1-2 (Desember)	Mencari dan menyempurnakan koding CDS yang masih kurang secara keseluruhan, kemudian mengembangkan CDS tambahan berdasarkan kebutuhan laporan BW untuk disimpan di S/4HANA W. Menyusun daftar lengkap CDS yang akan dibuat dari BW, mengecek kembali speed run CDS setelah penambahan view baru, serta mendiskusikan hasil performa dan rencana optimasi lanjutan dengan kepala tim agar landscape CDS siap digunakan untuk pengembangan laporan berikutnya.

3.3 Uraian Pelaksanaan Kerja

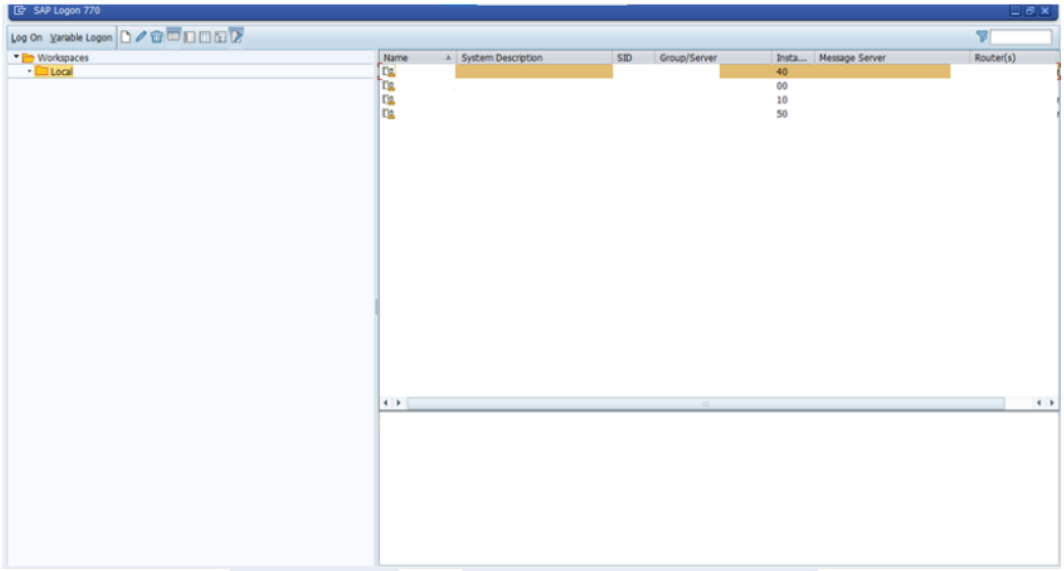
Gambar 3.3 Logo SAP

Sumber: commons.wikimedia.org (2025)

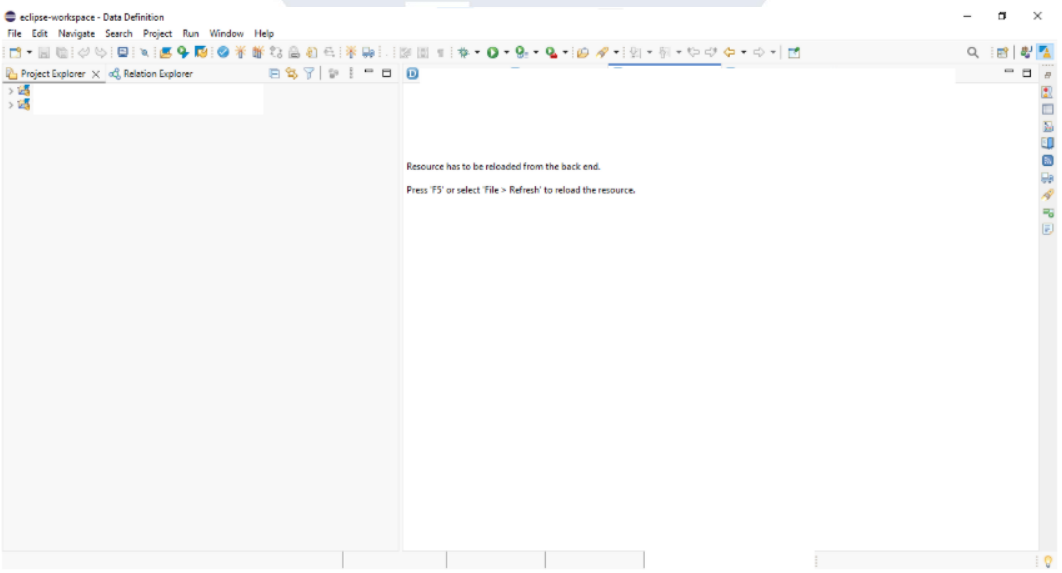
Gambar 3.4 Logo Eclipse

Sumber: logotyp.us

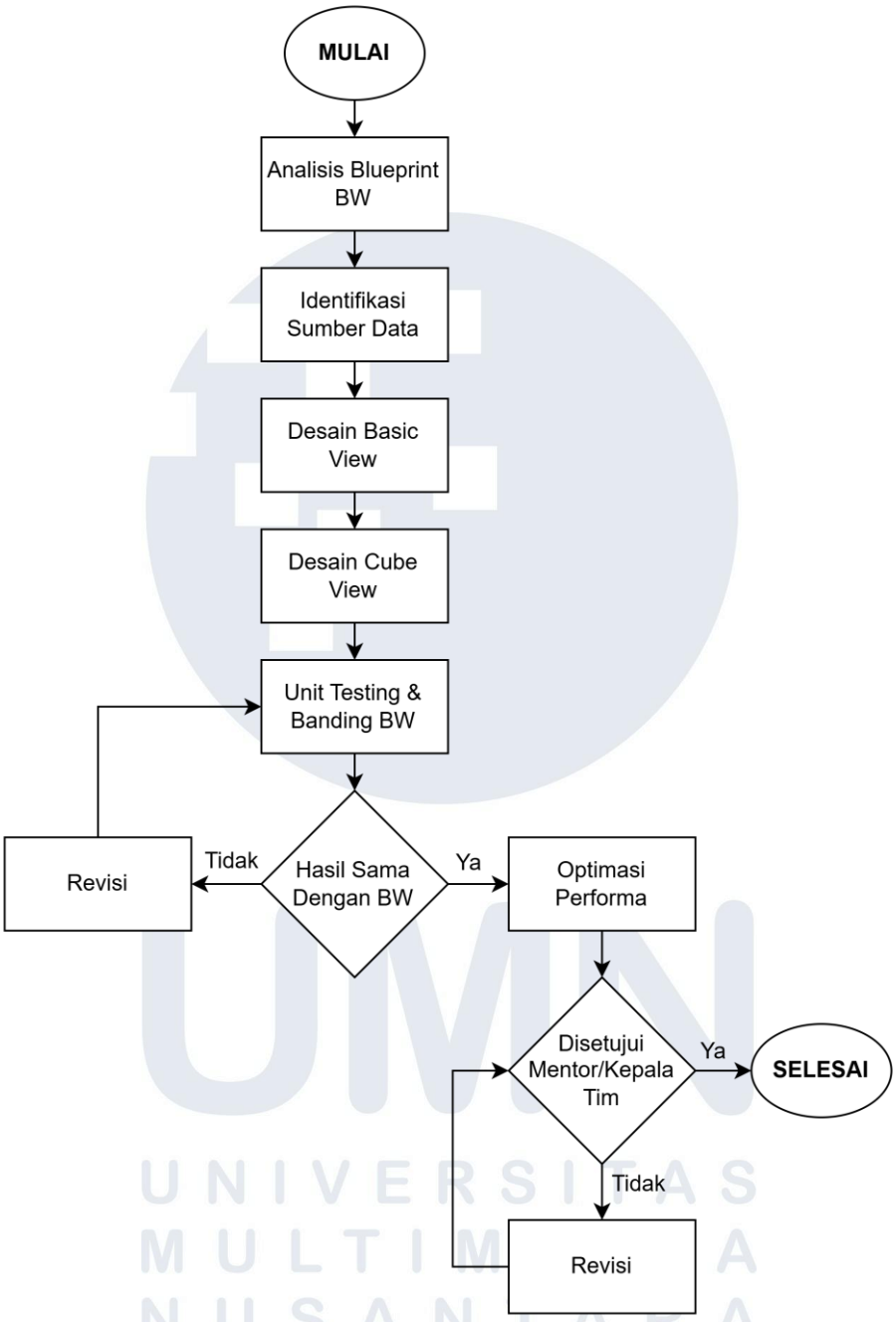
Secara umum, pekerjaan yang dilakukan selama kerja praktik merupakan satu rangkaian proyek besar, yaitu migrasi dan pemodelan ulang data SAP Business Warehousing (BW) ke ABAP Core Data Services (CDS) pada SAP S/4HANA untuk kebutuhan pelaporan PT X. Seluruh aktivitas dikerjakan dengan cara login ke sistem melalui SAP Logon, kemudian melakukan pengembangan CDS View di Eclipse (ABAP Development Tools).



Gambar 3.5 Logon Page SAP



Gambar 3.6 Page Awal Eclipse



Gambar 3.7 Diagram Alur Teknis Migrasi Data

Berikut ini adalah uraian proses pelaksanaan kerja yang dikelompokkan ke dalam beberapa bagian (*proyek/karya*) dari awal sampai tahap optimasi.

3.3.1 Proses Pelaksanaan

3.3.1.1 Proyek 1 - Pengenalan Lingkungan Kerja & Analisis Blueprint BW

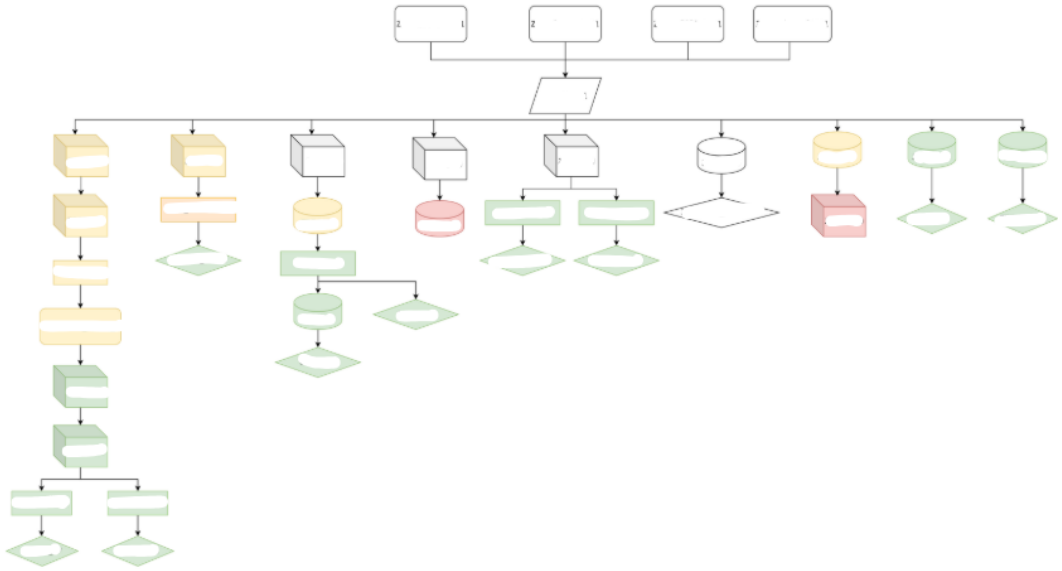
Pada tahap awal, lingkungan sistem yang digunakan dalam proyek diperkenalkan terlebih dahulu. Hal-hal yang dilakukan:

1) Akses Sistem

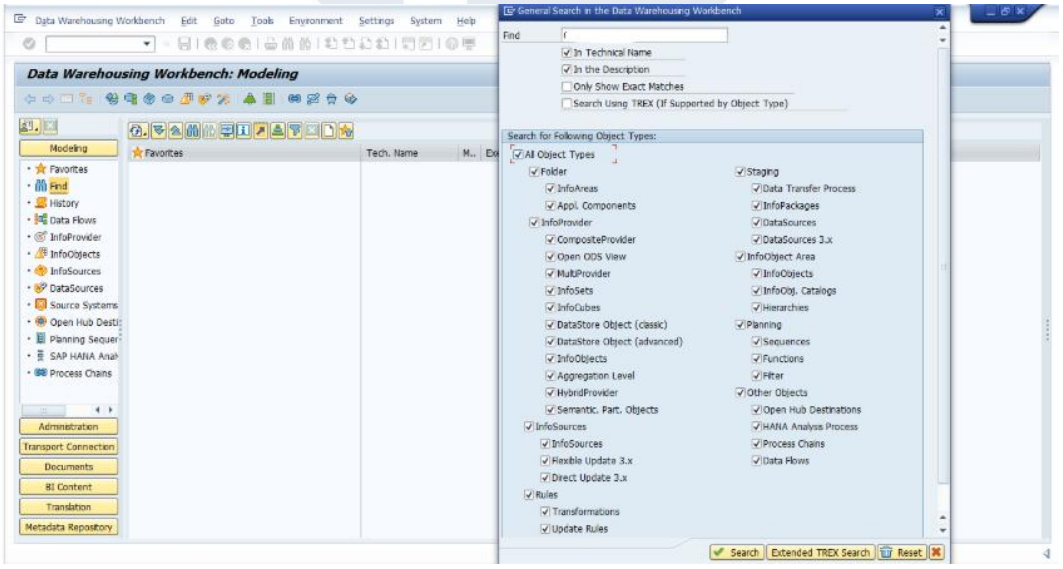
a) Masuk ke sistem SAP S/4HANA menggunakan SAP Logon dengan user yang telah disiapkan oleh tim.
b) Membuka Eclipse (ABAP Development Tools) sebagai *main tools* untuk pengembangan CDS View, termasuk pengaturan koneksi ke sistem SAP dan *workspace* proyek.

2) Pengenalan Blueprint Mapping BW

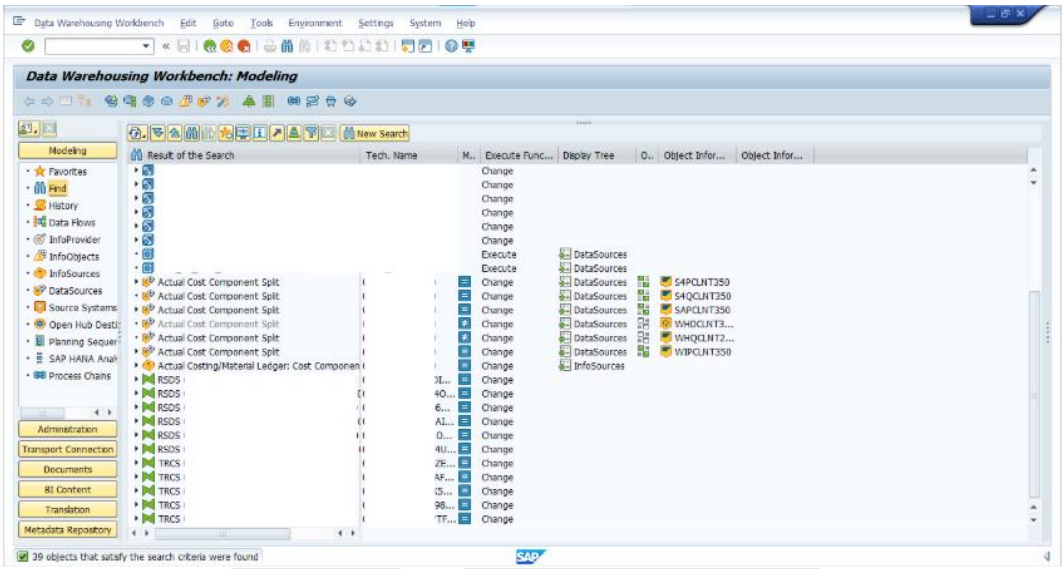
a) Mentor memberikan dokumen blueprint dan mapping BW dari PT X. Dokumen ini berisi struktur InfoProvider/Query di BW, daftar field, *key figure*, *dimension*, serta aturan-aturan perhitungan yang sudah berjalan.
b) Blueprint tersebut dipelajari untuk memahami:
(1) Sumber data utama (InfoCube/DSO),
(2) Hubungan antar-tabel/fakta dan master data
(3) Kebutuhan pelaporan (misalnya kombinasi filter, hierarki, dan agregasi yang harus didukung).



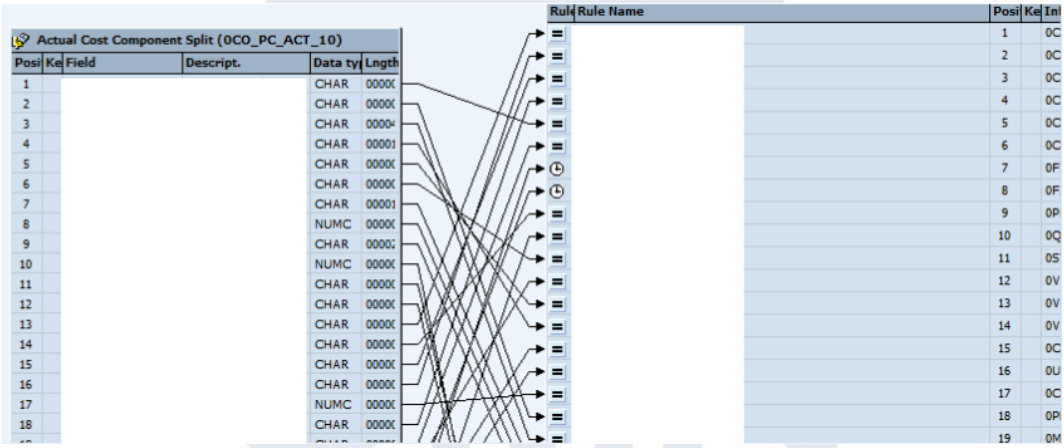
Gambar 3.8 Blueprint BW X



Gambar 3.9 Page Find Object BW



Gambar 3.10 Data Warehousing Bench BW



Gambar 3.11 Contoh Struktur antar Object BW

3) Analisis Kebutuhan CDS View

a) Bersama mentor, bagian-bagian blueprint BW yang harus dipertahankan di lingkungan S/4HANA, dan bagian yang dapat disederhanakan ketika dimigrasikan ke CDS diidentifikasi.

b) Dari hasil analisis disusun daftar awal:

(1) field yang wajib ada di CDS,

(2) kandidat sumber data (I_View dan tabel),

(3) serta kemungkinan pemanfaatan CDS basic view, cube view, dan consumption view.

Fase pertama merupakan tahap pemahaman menyeluruh terhadap struktur data SAP BW yang telah digunakan PT X. Pada tahap ini, analisis terhadap InfoProvider seperti DataStore Object (DSO), InfoCube, dan Query BW dilakukan. Aktivitas analisis dilakukan melalui:

1) Pembacaan field karakteristik (characteristics) dan key figure untuk memahami struktur data yang dilaporkan.
2) Menelusuri master data (GL Account, Cost Center, Profit Center, Material, WBS) yang berperan sebagai pengayaan laporan.
3) Membaca transformation rule untuk memahami bagaimana BW melakukan kalkulasi, mapping, dan relasi antar tabel.
4) Membuat dokumen mapping awal yang memetakan field BW ke I_View standar SAP S/4HANA.

Tahap analisis ini menjadi fondasi penting karena menentukan field mana yang harus dimigrasikan, logika mana yang harus dipertahankan, serta struktur CDS apa saja yang akan dibangun.

Output Fase 1:

1) Dokumen mapping awal BW -> CDS
2) Daftar field wajib & optional
3) Identifikasi key figure, dimension, dan parameter yang diperlukan

3.3.1.2 Proyek 2 - Perancangan CDS View Awal Berdasarkan Blueprint

Setelah pemahaman dasar diperoleh, tahap perancangan CDS View berdasarkan blueprint BW mulai dimasuki. Langkah-langkah utama:

1) Pencarian Sumber I_View

a) I_View standar SAP (misalnya I_ActualPlanJrnlEntryItem, I_MaterialDocumentItem, dsb.) yang relevan dengan field pada blueprint BW ditelusuri.

b) Proses ini dilakukan dengan memanfaatkan fasilitas pencarian di Eclipse, serta referensi dari mentor untuk memahami konvensi penamaan I_View.

c) Setiap field pada blueprint dicocokkan dengan asal data di I_View atau tabel yang sesuai, kemudian didokumentasikan dalam bentuk mapping.

2) Join antar I_View dan Master Data

a) Setelah sumber data ditemukan, CDS basic view dan cube view dengan melakukan join antar I_View serta master data dibangun (misalnya tabel atribut cost center, GL, profit center, WBS, dan lain-lain).

b) CDS association untuk menghubungkan fakta dengan master data dimanfaatkan, sehingga struktur menjadi lebih modular dan mudah dipelihara.

3) Penerapan Function, Formula, dan Filter

a) Pada blueprint BW terdapat berbagai rumus/key figure dan filter condition yang sebelumnya ditanamkan di query BW.

b) Rumus-rumus tersebut dianalisis (misalnya perhitungan varians, rasio, dan indikator tertentu) kemudian menerjemahkannya ke

dalam ekspresi di CDS (menggunakan statement CASE, SUM, CAST, dsb.), atau jika kompleks diletakkan di CDS Table Function.

c) Filter dasar (misalnya company code tertentu, fiscal year, atau jenis transaksi) diimplementasikan dalam bentuk parameter CDS dan/atau kondisi pada WHERE clause.

```
@AbapCatalog.viewEnhancementCategory: [#NONE]
@AccessControl.authorizationCheck: #NOT_REQUIRED
@EndUserText.label: '
@Analytics.dataCategory: #CUBE
@Metadata.ignorePropagatedAnnotations: true
define view entity
  as select from    I_ActualPlanJrnlEntryItemCube as APJ
    left outer join I_CompanyCode                 as CoCd  on CoCd.CompanyCode = APJ.CompanyCode
    left outer join I_ControllingArea             as COAr  on COAr.ControllingArea = APJ.ControllingArea
    left outer join I_CostCenter                  as CCTR  on  CCTR.ControllingArea = APJ.ControllingArea
                                                           and CCTR.CostCenter      = APJ.CostCenter
    left outer join I_GLAccountInChartOfAccounts  as GL    on  GL.ChartOfAccounts = APJ.ChartOfAccounts
                                                           and GL.GLAccount       = APJ.GLAccount
    left outer join I_FunctionalArea              as FA    on FA.FunctionalArea = APJ.FunctionalArea
    left outer join I_FunctionalArea              as PFA   on PFA.FunctionalArea = APJ.PartnerFunctionalArea
    left outer join I_FinancialManagementArea     as FMA   on FMA.FinancialManagementArea = APJ.FinancialManagementArea
    left outer join I_Fund                        as FUND  on FUND.Fund = APJ.Fund
    left outer join I_Fund                        as PFUND on PFUND.Fund = APJ.PartnerFund
    left outer join I_Grant                       as GR    on GR.GrantID = APJ.GrantID
    left outer join I_Grant                       as PGR   on PGR.GrantID = APJ.PartnerGrant

{
  APJ.FiscalYearPeriod,
  APJ.FiscalYearVariant          as FiscalYearVariant,
  APJ.FiscalYear,
  APJ.FiscalPeriod,
  APJ.CostCenter,
  APJ.CompanyCode,
  APJ.ControllingArea,
  APJ.PartnerCostCenter,
  APJ.ChartOfAccounts,
  APJ.GLAccount,
  APJ.FinancialManagementArea    as FMArea,         // FIKRS
  APJ.FunctionalArea             as FunctionalArea, // FKBER
  APJ.Fund                       as Fund,           // GEBER
  APJ.GrantID, // GRANT_NBR
  APJ.PartnerFunctionalArea      as PartnerFunctionalArea, // PFKBER
  APJ.PartnerFund                as PartnerFund, // PGEBER
  APJ.PartnerGrant               as PartnerGrant. // PGRANT_NBR
```

Gambar 3.12 Contoh Kodingan Join i_ Awal

```
where
      zf.DocumentType                 !=      'CO'
  and(
      zf.CostElement                  =       '0033100005'
    or zf.CostElement                 =       '0065000001'
    or zf.CostElement                 between '0051101001' and '0051900099'
    or zf.CostElement                 between '0052101001' and '0052900099'
    or zf.CostElement                 between '0071101001' and '0071900099'
  )
  and(
      zf.DebitAmountInLocalCurrency   !=      0
    or zf.CreditAmountInLocalCurrency !=      0
    or zf.AmountInLocalCurrency       !=      0

  )
```

Gambar 3.13 Code Filter di CDS

Rule Details

Pattern | Pretty Printer | Routines Info.

```
180          CLEAR : lv_costcenter.
181
182          IF SOURCE_FIELDS-wbs_elemt(1) EQ 'C'.
183 *          lv_wbs = SOURCE_FIELDS-wbs_elemt(8).
184 *          CONCATENATE lv_wbs '00' INTO lv_wbs
185 *            SEPARATED BY '  '.
186            lv_wbs = SOURCE_FIELDS-wbs_elemt+5(3).
187            IF lv_wbs EQ 'PRO'.
188              lv_wbs = lv_wbselemt_external.
189            ELSE.
190              lv_wbs = SOURCE_FIELDS-wbs_elemt(8).
191              CONCATENATE lv_wbs '00'
192              INTO lv_wbs
193              SEPARATED BY '  '.
194            endif.
195          ELSEIF SOURCE_FIELDS-wbs_elemt(1) EQ 'J'.
196            lv_wbs = SOURCE_FIELDS-wbs_elemt(11).
197            CONDENSE lv_wbs.
198          ELSEIF SOURCE_FIELDS-wbs_elemt(1) EQ 'S'.
199            lv_wbs = SOURCE_FIELDS-wbs_elemt(10).
200            CONDENSE lv_wbs.
201          ELSEIF SOURCE_FIELDS-wbs_elemt(1) EQ 'M'.
202            lv_wbs = SOURCE_FIELDS-wbs_elemt(15).
203            CONCATENATE lv_wbs '00' INTO lv_wbs.
204            CONDENSE lv_wbs.
205          ELSEIF SOURCE_FIELDS-wbs_elemt(1) EQ 'B' OR
```

Gambar 3.14 Function/Rumus Field di BW

```
b.client
,b.ControllingArea
,b.CostCenter
,b.DocumentType
,b.Plant
,b.ProfitCenter
,b.ReferenceKey
,b.WBSElement
,case
    when ltrim(rtrim(b.ProfitCenter)) IN ('3998', LPAD('3998',10,'0')) THEN
        CASE
            when b.ProfitCenterConv in( '' ,'0') and substring( b.WBSElement,1,1 ) ='P' then
                case
                    when substring( b.WBSElement,7,3 ) ='999'
                        then '3999'
                        else case
                                when ltrim(rtrim(REPLACE_REGEXPR(' +' IN ztarayon.Sales_office with ' '))) = '3999'
                                    THEN LTRIM(RTRIM(REPLACE_REGEXPR(' +' IN ztarayon.Sales_office with ' ')))
                                WHEN ltrim(rtrim(REPLACE_REGEXPR(' +' IN ztarayon.Sales_office with ' '))) between '300
                                    THEN LTRIM(RTRIM(REPLACE_REGEXPR(' +' IN ztarayon.Sales_office with ' ')))
                                ELSE
                                    CONCAT(SUBSTRING(LTRIM(RTRIM(REPLACE_REGEXPR(' +' IN ztarayon.Sales_office with ' '
                                END
                END
        END
    WHEN substring(b.WBSElement,1,1) ='M'
    THEN
        CASE
            when LTRIM(RTRIM(REPLACE_REGEXPR(' +' IN ztarayon2.Sales_office with ' '))) = '3999'
                THEN LTRIM(RTRIM(REPLACE_REGEXPR(' +' IN ztarayon2.Sales_office with ' ')))
            WHEN ltrim(rtrim(REPLACE_REGEXPR(' +' IN ztarayon2.Sales_office with ' '))) between '3001' and '3010'
                THEN LTRIM(RTRIM(REPLACE_REGEXPR(' +' IN ztarayon2.Sales_office with ' ')))
            ELSE
                CONCAT(SUBSTRING(LTRIM(RTRIM(REPLACE_REGEXPR(' +' IN ztarayon2.Sales_office with ' '))),1,3),'0')
            END
    ELSE
```

Gambar 3.15 Function di Amdp CDS

CDS tingkat dasar (basic view) dibangun dengan fokus pada penentuan sumber data dan struktur join. Aktivitas pada fase ini meliputi:

1) Identifikasi I_View dan tabel relevan, seperti:

a) I_ActualPlanJrnlEntryItem
b) I_MaterialDocumentItem
c) I_CompanyCode
d) I_ProfitCenter
e) Tabel master data tambahan (GLA, CSKS, CEPC, dll.)

2) Penyusunan CDS Basic View dengan melakukan:

a) Pemilihan field yang relevan.
b) Pemberian alias yang konsisten dengan standar W.
c) Penentuan kunci join yang tepat untuk mencegah penggandaan data.

3) **Pembuatan CDS Cube View untuk menggabungkan fakta dan master data melalui association.**
4) **Penerapan parameter seperti fiscal year, period, company code untuk memberikan fleksibilitas pada konsumsi laporan.**

Pada tahap ini logika BW ke logika CDS menggunakan fungsi-fungsi mulai diterjemahkan seperti CASE, SUM, CAST, serta filter pada WHERE clause.

Output Fase 2:

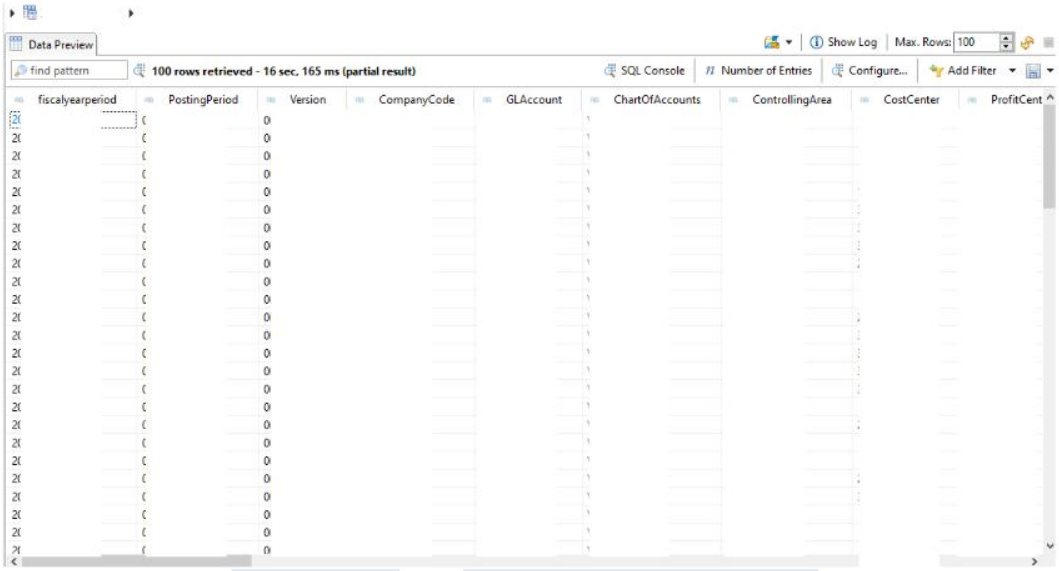
1) Basic View versi awal
2) Cube View versi awal
3) Daftar join dan sumber data yang terkonfirmasi
4) Parameter CDS yang telah disepakati

3.3.1.3 Proyek 3 - Mengikuti Mockup, Blueprint CDS, dan Penyempurnaan Struktur Join

Pada tahap ini, fokus diarahkan pada blueprint CDS dan *mockup* laporan.

1) Penyusunan Mockup dan Blueprint CDS

a) Berdasarkan hasil analisis dan percobaan awal, mockup tampilan laporan (misalnya layout kolom, baris, filter) yang akan dihasilkan dari CDS disusun.
b) Mockup ini kemudian diturunkan menjadi blueprint CDS yang memuat daftar view (basic, cube, consumption), hubungan antar view, serta alur data dari tabel sumber hingga laporan akhir.



Gambar 3.16 Contoh Output CDS

2) Diskusi dan Review dengan Kepala Tim

a) Mockup dan blueprint CDS dipresentasikan untuk mendapatkan konfirmasi apakah desain sudah sesuai kebutuhan bisnis PT X.

b) Dari sesi ini masukan terkait penyederhanaan struktur join, pengelompokan field, serta penyesuaian nama-nama elemen diperoleh agar konsisten dengan standar penamaan di W.

3) Penyempurnaan Join dan Struktur View

a) Berdasarkan umpan balik tersebut, revisi CDS yaitu menghapus join yang tidak diperlukan, merapikan alias dan nama field, serta memastikan bahwa setiap join memiliki kunci yang jelas untuk menghindari multiplikasi data dilakukan.

```
@Semantics.amount.currencyCode: 'LocalCurrency'
zf.DebitAmountInLocalCurrency  as TotalDebitPosting,
@Semantics.amount.currencyCode: 'LocalCurrency'
zf.CreditAmountInLocalCurrency as TotalCreditPosting,
@Semantics.amount.currencyCode: 'LocalCurrency'
zf.AmountInLocalCurrency       as CumulativeBalance,
zf.postingdateinthedocument    as CustomDate,
```

Gambar 3.17 Contoh Aliasing

```
@AbapCatalog.viewEnhancementCategory: [#NONE]
@AccessControl.authorizationCheck: #NOT_REQUIRED
@EndUserText.label: 'Cost and Allocation (Actual & Budget)'
@Metadata.ignorePropagatedAnnotations: true
@ObjectModel.usageType:{
    serviceQuality: #X,
    sizeCategory: #S,
    dataClass: #MIXED
}
define view entity ZC_
  with parameters
    p_FiscalYear : abap.numc(4)
  as select from ZC_          (p_FiscalYear:$parameters.p_FiscalYear)
{
  CostCenter                    as Department,
  CompanyCode                   as Company,
  FiscalYear                    as FiscalYear,
  PostingPeriod                 as PostingPeriod,
  PostingPeriod                 as RecapPeriod,
  CostElementMapping            as CostElementMapping,
  cast('' as abap.char(18))     as Assignment,
  CalendarYear                  as CalendarYear,
  cast( '00000' as abap.numc(5) ) as CalYearQuarter,
  cast( '0' as abap.numc(1) )   as CalQuarter,
  FiscalYearVariant             as FiscalYearVariant,
  cast('' as abap.cuky( 5 ))    as LocalCurrency,
  @Semantics.amount.currencyCode: 'LocalCurrency'
  cast(0 as abap.curr( 23, 2 )) as Debit,
  @Semantics.amount.currencyCode: 'LocalCurrency'
  cast(0 as abap.curr( 23, 2 )) as Credit,
  @Semantics.amount.currencyCode: 'LocalCurrency'
  cast(0 as abap.curr( 23, 2 )) as Amount,
  cast('' as abap.numc(3))      as ValueType,
  cast('' as abap.char(60))     as ItemText,
  cast('' as abap.char(24))     as WBSElement,
  cast('' as abap.char(24))     as ZWBS,
  cast('' as abap.char(1))      as ProjectTypeWismilak,
  cast('' as abap.char(10))     as DocNumberGeneralLedgerView,
  cast('' as abap.char(10))     as GLAccount,
```

Gambar 3.18 Contoh Code CDS

Menyusun blueprint CDS sebagai desain arsitektural lengkap agar seluruh view memiliki arah yang seragam. Kegiatan pada fase ini antara lain:

1) Penyusunan Mockup Laporan

Menyiapkan visualisasi awal bentuk laporan seperti:

a) Kolom yang ditampilkan.
b) Filter yang disediakan.
c) Struktur baris dan keterkaitannya dengan indikator bisnis.

2) Pembuatan Blueprint CDS Berlapis

Blueprint menjelaskan struktur:

a) Basic View: berisi fakta dan field dasar
b) Cube View: field agregasi, perhitungan, key figure
c) Consumption View: level yang dikonsumsi Fiori/SAC

3) Review bersama Mentor & Kepala Tim

Pada fase ini struktur CDS dipresentasikan untuk mendapatkan masukan terkait:

a) Efisiensi join.
b) Konsistensi penamaan field.
c) Keberlanjutan maintenance jangka Panjang.
d) Kesesuaian dengan kebutuhan pelaporan PT X.

4) Revisi desain berdasarkan masukan

Struktur join diperbaiki, menyederhanakan field, dan memastikan blueprint selaras dengan standar desain W.

Output Fase 3:

1) Blueprint CDS final
2) Mockup laporan terverifikasi
3) Struktur view yang rapi dan konsisten
4) Hasil review mentor dan perbaikan desain

3.3.1.4 Proyek 4 - Redesain Struktur Mapping CDS dan Penyederhanaan Arsitektur

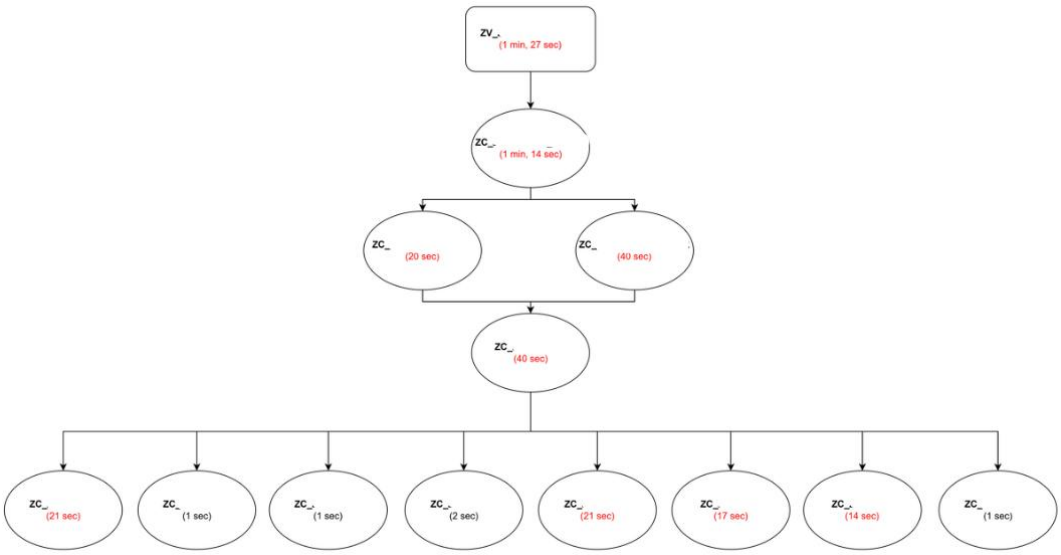
Di tengah masa kerja praktik, tim menemukan bahwa struktur join yang awal masih menimbulkan masalah, terutama terkait kesulitan join dan kompleksitas di level atas. Pada fase ini, struktur mapping dilakukan redesain.

1) Identifikasi Permasalahan Struktur Awal

Saat melakukan uji coba dan debugging, ditemukan bahwa beberapa data tidak dapat ter-join dengan benar sehingga menghasilkan angka yang tidak konsisten. Struktur CDS di bagian atas terlalu banyak menggunakan join dan union, sehingga logika menjadi sulit ditelusuri dan performa menurun.

2) Pendekatan Desain Baru

Pendekatan baru Join dilakukan sejak level awal dengan sumber data yang benar-benar diperlukan, dan hanya field-field yang relevan yang diambil ke view berikutnya, sehingga CDS di level atas (cube/consumption) menjadi lebih ramping dan mudah dibaca. Mapping ini disusun ulang dalam bentuk struktur mapping CDS baru, yang kemudian dibandingkan dengan struktur mapping BW untuk memastikan seluruh kebutuhan tetap tercakup.



Gambar 3.19 Struktur Blueprint Baru CDS

3) Implementasi dan Uji Konsistensi

Desain baru di Eclipse diimplementasikan, memecah view yang terlalu besar, serta mengatur ulang association dan parameter. Selanjutnya dilakukan perbandingan hasil antara laporan BW dan CDS yang baru untuk memastikan bahwa data dan perhitungannya konsisten.

Ketika proses mulai memasuki tahap implementasi lanjutan, ditemukan sejumlah masalah pada struktur CDS versi awal, yaitu:

a) Join yang terlalu banyak dan kompleks.
b) Data yang tidak terhubung (unmatched records).
c) Data gandaan (duplicates).
d) Kesalahan panjang field pada UNION.
e) Performa eksekusi yang lambat.

Struktur CDS dilakukan redesain penuh, dengan pendekatan sebagai berikut:

Prinsip Redesain Arsitektur:

a) Join dilakukan sedini mungkin di basic view agar cube view lebih ringan.
b) Hanya field penting yang diteruskan ke layer atas untuk mengurangi overhead.
c) Memecah union berdasarkan keseragaman field untuk menghindari activation error.
d) Menghapus join tak perlu, misalnya join terhadap lookup data yang tidak digunakan.
e) Standarisasi key figure dan tipe data untuk menjaga konsistensi.
f) Mendesain ulang mapping dalam bentuk struktur berlapis yang lebih mudah ditelusuri.

Tahap redesain ini merupakan salah satu kontribusi terbesar dalam proyek karena membantu membuat CDS menjadi lebih stabil, cepat, dan mudah dirawat.

Output Fase 4:

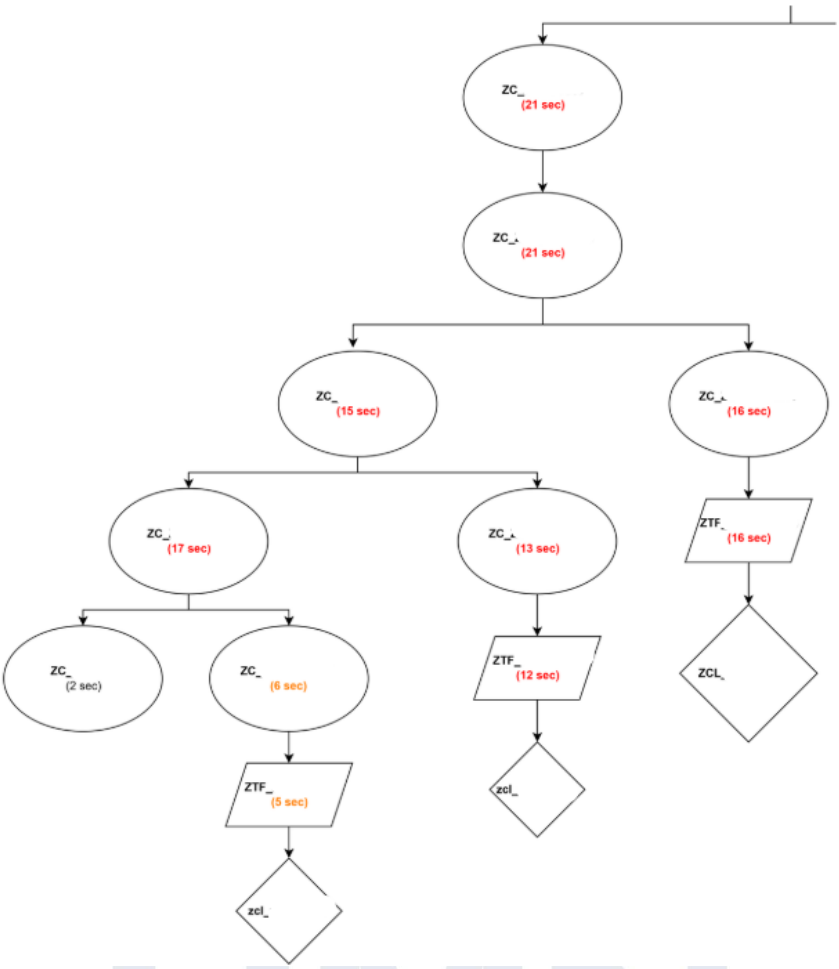
1) Arsitektur CDS baru yang lebih efisien
2) Berkurangnya duplicate data
3) Join lebih terstruktur dan mudah dipelihara
4) Aktivasi view lebih stabil tanpa error panjang field

3.3.1.5 Proyek 5 - Pengujian Performa, Konversi AMDP ke CDS, dan Penanganan Duplicate Data

Pada fase akhir, fokus pekerjaan beralih ke optimasi performa dan kualitas data.

1) Pengujian Performa (Speed Test)

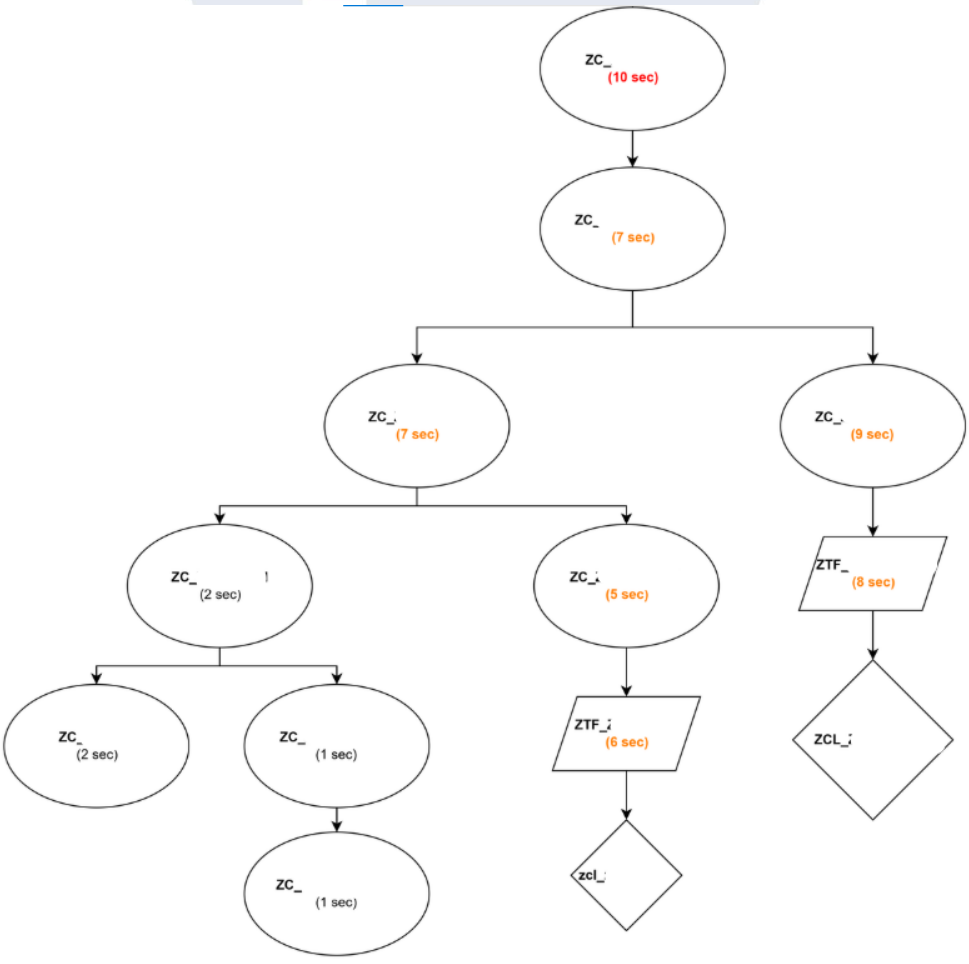
CDS query dijalankan dan waktu eksekusi dicatat untuk berbagai kombinasi parameter (misalnya periode, company code, dan level detail). Hasil pengujian menunjukkan bahwa beberapa bagian masih memerlukan waktu eksekusi yang relatif lama.



Gambar 3.20 Bagian Struktur CDS dengan Speed Run

2) Konversi AMDP Table Function ke CDS

Di dalam arsitektur sebelumnya terdapat beberapa AMDP Table Function yang digunakan untuk melakukan perhitungan tertentu. Untuk mengurangi overhead dan memanfaatkan kemampuan optimasi CDS, menganalisis logika di tiap AMDP, mengonversinya menjadi CDS Table Function atau ekspresi murni di CDS jika memungkinkan, dan mengintegrasikannya kembali ke dalam rantai view yang sudah ada. Setelah konversi, dilakukan pengujian ulang dan terbukti terdapat peningkatan kecepatan run dibandingkan struktur sebelumnya.



Gambar 3.21 Speed Setelah di Optimasi

3) Penanganan Duplicate Data

Pada saat validasi data, ditemukan adanya duplicate record setelah proses konversi dan join baru. Analisis bersama mentor dilakukan untuk mengidentifikasi penyebabnya, misalnya join yang belum cukup spesifik, kombinasi kunci yang belum lengkap, atau data sumber yang secara bisnis memang multi-line. Solusi yang diambil antara lain: memperbaiki kondisi join, menambahkan grouping dan agregasi yang tepat, serta melakukan pengecekan record count antara BW dan CDS sebagai kontrol.

Fase terakhir berfokus pada peningkatan performa dan penjaminan kualitas data. Proses optimasi dilakukan melalui:

a) **Pengujian performa (runtime test)**

CDS dijalankan dengan parameter berbeda untuk mencatat waktu eksekusi sebelum dan sesudah redesain.

b) **Konversi AMDP -> CDS Table Function**

AMDP yang sebelumnya digunakan untuk perhitungan kompleks diganti menjadi:

(1) CDS Table Function (jika perlu SQLScript), atau
(2) murni CDS expression (jika logikanya sederhana). Konversi ini dilakukan karena CDS dapat lebih optimal memanfaatkan in-memory SAP HANA.

c) **Penanganan Duplicate Data**

Dengan memeriksa kunci join, level agregasi, dan kondisi UNION, duplicate berhasil dihilangkan.

d) **Validasi Akhir dengan Laporan BW**

Perbandingan angka dilakukan terhadap laporan BW sebagai referensi utama.

e) **Perbaikan final berdasarkan temuan validasi**

Perbaikan terjadi pada beberapa field key figure, tanggal posting, serta struktur kumpulan data tertentu.

Output Fase 5:

1) Performa CDS meningkat signifikan
2) AMDP berhasil dihilangkan atau dikurangi
3) Konsistensi data BW vs CDS tercapai
4) Model CDS siap digunakan untuk analitik modern (Fiori/SAC)

3.3.1.6 Proyek 6 - Pengujian Performa, Konversi AMDP ke CDS, dan Penanganan Duplicate Data

Pada fase akhir masa kerja praktik, fokus pekerjaan bergeser pada proses finalisasi dan penambahan CDS tambahan yang akan disimpan dan digunakan di lingkungan S/4HANA milik PT W (W). Fase ini mencakup penyempurnaan koding, pembuatan CDS tambahan berdasarkan kebutuhan BW, evaluasi performa keseluruhan, serta diskusi lanjutan dengan kepala tim mengenai kesiapan arsitektur CDS.

1) Penyempurnaan dan Perbaikan Koding CDS yang Sudah Ada

Di tahap awal minggu keempat November, dilakukan pencarian dan perbaikan terhadap seluruh struktur CDS yang sebelumnya telah dibangun. Tujuan kegiatan ini adalah memastikan bahwa:

a) Logika perhitungan telah konsisten dan tidak terjadi perbedaan angka dengan referensi BW.
b) Join dan association yang tidak relevan dihapus untuk mengurangi beban runtime.
c) Alias, naming convention, tipe data, dan key figure telah distandarkan agar sesuai praktik terbaik W.
d) Kesalahan minor seperti field yang tidak terpakai, annotation yang belum lengkap, atau struktur yang belum rapi diperbaiki secara menyeluruh.

Tahap ini sangat penting untuk memastikan bahwa seluruh CDS memiliki kualitas yang layak sebelum disimpan permanen di S/4HANA W.

2) Pengembangan CDS Tambahan Berdasarkan Kebutuhan Pelaporan BW

Setelah proses penyempurnaan, tim memberikan tugas lanjutan untuk membuat CDS tambahan yang diperlukan untuk:

a) Replikasi laporan-laporan BW tertentu yang belum lengkap di fase sebelumnya.
b) Penyediaan data tambahan bagi tim analitik dan reporting di W.

Dalam fase ini, dilakukan:

a) Penelusuran ulang blueprint BW terkait InfoCube atau InfoObject yang belum ditransformasikan.
b) Identifikasi asal field dari berbagai I_View standar SAP.
c) Penyusunan basic view, cube view, dan consumption view baru (jika diperlukan).
d) Penyesuaian key, aggregation, dan semantic annotation agar data mudah dikonsumsi oleh layer analytics.

Beberapa CDS tambahan ini nantinya akan digunakan langsung oleh tim W untuk pengembangan laporan lanjutan.

3) Pengujian Performa (Speed Run) dan Evaluasi Stabilitas View

Setelah CDS tambahan selesai dibuat, dilakukan speed run test untuk mengukur performa keseluruhan dan memastikan view baru tidak memperlambat request.

Hasil evaluasi mencakup:

a) Waktu eksekusi masing-masing view dibandingkan dengan versi sebelumnya.
b) Identifikasi bagian yang masih menimbulkan bottleneck.
c) Deteksi kemungkinan join berat, kalkulasi kompleks, atau aggregation yang dapat dioptimalkan.
d) Uji konsistensi setelah penambahan CDS baru untuk memastikan tidak muncul duplicate records, missing join, atau mismatch angka BW.

Jika ditemukan isu pada fase ini, dilakukan perbaikan langsung sebelum proses finalisasi.

Secara menyeluruh, proses pelaksanaan kerja dalam proyek ini mengikuti metodologi pengembangan model data yang terstruktur. Setiap fase dilakukan secara iteratif melalui koordinasi dengan mentor dan kepala tim, sehingga hasil CDS yang dibangun tidak hanya memenuhi kebutuhan pelaporan PT X, tetapi juga mengikuti prinsip best practice SAP S/4HANA dan SAP BTP. Berikut adalah uraian lengkap proses pelaksanaan dalam lima fase utama.

3.3.2 Kendala yang Ditemukan

Selama pelaksanaan proyek migrasi dari SAP BW ke ABAP CDS dan AMDP, sejumlah kendala yang bersifat konseptual maupun teknis dihadapi. Kendala-kendala ini muncul terutama karena perbedaan cara kerja antara BW, CDS, dan AMDP, serta kompleksitas struktur data dan logika pelaporan yang sudah berjalan di sistem lama.

1) Kompleksitas Struktur SAP BW yang Sulit Ditelusuri

Banyak laporan di BW dibangun dari lapisan-lapisan objek (DSO, InfoCube, Query) yang saling terkait. Akibatnya, ketika dari mana asal suatu angka coba dipahami, prosesnya tidak bisa dilakukan hanya dengan satu atau dua tabel saja, tetapi harus menelusuri banyak lapisan.

Ditemukan bahwa Satu InfoProvider dapat memiliki beberapa transformation rule, lookup ke master data tambahan, dan bahkan rutin ABAP di dalam transformasi. Misalnya, sebuah key figure di Query BW bisa berasal dari kombinasi beberapa key figure DSO yang sebelumnya

sudah di-aggregate, lalu mengalami kalkulasi ulang (misalnya varians, rasio, atau proyeksi). Hal ini mempersulit proses *reverse engineering* karena harus:

a) Membuka Transformation dan Routine satu per satu.
b) Mengidentifikasi field sumber di tabel staging.
c) Menelusuri *lineage* data dari source system -> DSO -> InfoCube -> Query.

Ketika logika ini dipindahkan ke CDS, tidak ada fitur otomatis "show transformation" seperti di BW, sehingga seluruh alur harus dimodelkan ulang secara manual.

2) Join Antar Sumber Data Menyebabkan Duplicate dan Data Tidak Terhubung

Ketika beberapa sumber data digabungkan (join), kadang jumlah baris justru bertambah banyak atau sebagian data hilang. Ini menyebabkan angka laporan menjadi lebih besar/lebih kecil dari seharusnya.

Kendala muncul saat:

a) Join dilakukan hanya berdasarkan primary key yang tidak lengkap.
b) Tabel fakta di-join dengan tabel lain yang memiliki multiple record per key (many-to-many).
c) Join tidak disertai filter atau kondisi tambahan.

Contoh kasus:

a) Join antara tabel jurnal (ledger) dengan cost center tanpa memperhatikan valid-from/valid-to.

b) Join antara material document dengan tabel agregasi tanpa kunci unik tambahan.

Akibatnya:

a) Terjadi data multiplication (misalnya 1 baris transaksi menjadi 2–3 baris).
b) Beberapa baris menjadi unmatched karena tidak ada pasangan key di sisi lain.
c) Hasil agregasi SUM menjadi tidak valid.

3) Error UNION karena Ketidaksesuaian Struktur Field

CDS mengharuskan semua kolom pada UNION memiliki tipe data dan panjang yang sama. Jika ada sedikit perbedaan, view gagal di-activate dan mengeluarkan error yang cukup teknis. Contoh error yang paling sering ditemui karena perbedaan length.

Hal ini muncul ketika:

a) Satu sisi UNION menggunakan CHAR (18) sementara sisi lain CHAR(10).
b) Ada perbedaan DEC(15,2) dan DEC(23,2).
c) Field amount di satu sisi sudah di-cast, di sisi lain belum.

Karena CDS tidak melakukan auto-cast dalam UNION, harus dilakukan:

a) Mengidentifikasi semua field yang terlibat di UNION.
b) Memastikan length dan tipe data identic.
c) Menambahkan CAST eksplisit jika dibutuhkan.

Proses ini memakan waktu karena satu UNION bisa melibatkan banyak field.

4) Performa View Lambat dan Berat Dieksekusi

Pada beberapa versi awal desain, saat query dijalankan, waktu respon terasa lama. Hal ini tidak ideal untuk laporan operasional yang diharapkan mendekati real-time.

Penyebab performa lambat di antaranya:

a) Terlalu banyak join dalam satu view (termasuk join yang tidak kritikal).
b) Layer CDS terlalu "gemuk" (basic view mengambil terlalu banyak field, cube juga memproses semua field).
c) Beberapa perhitungan (CASE, agregasi kompleks) dilakukan di level tinggi yang sudah mengandung banyak data.

Karena CDS dijalankan di atas SAP HANA, desain view yang tidak efisien akan membuat *query plan* menjadi kompleks dan menghabiskan resource (CPU dan memory) lebih dari yang diperlukan.

5) Struktur AMDP yang Rumit dan Sulit Dipelihara

Sebagian logika awal dibuat dalam bentuk AMDP Table Function. Namun, struktur AMDP yang panjang membuatnya sulit dibaca, dipelihara, dan di-debug, terutama bagi intern yang baru pertama kali terpapar SQLScript.

AMDP:

a) Menggunakan SQLScript dengan sintaks yang berbeda dari ABAP biasa.

b) Berjalan di database layer, sehingga debugging membutuhkan pendekatan berbeda.
c) Sering berisi banyak nested SELECT, subquery, dan conditional logic.

Kendala yang dihadapi:

a) Sulit memetakan satu bagian AMDP terhadap bagian tertentu di BW.
b) Susah memastikan titik mana yang menyebabkan performa turun.
c) Perubahan kecil kadang mempengaruhi keseluruhan struktur.

Hal ini membuat pemindahan sebagian logika kembali ke CDS perlu dipertimbangkan.

6) Menerjemahkan Function BW ke AMDP Table Function

BW menyediakan *formula editor* dan fungsi built-in yang "tinggal pakai", sedangkan AMDP mensyaratkan semua logika ditulis manual dalam bentuk SQLScript. Tidak semua fungsi BW memiliki padanan langsung, sehingga sering kesulitan menemukan cara menerjemahkannya.

Misalnya:

a) IF/ELSE BW harus diubah menjadi CASE WHEN atau IF SQLScript.
b) Exception aggregation BW harus dipecah menjadi GROUP BY & window function,
c) Restricted key figure perlu ditulis sebagai subquery dengan filter tertentu.
d) Formula yang tadinya satu baris di BW bisa menjadi beberapa baris SQLScript.

Selain itu, ada konsep seperti:

a) *Calculate result as* di BW yang tidak otomatis ada di AMDP.
b) Penggunaan variable dan filter BW yang tidak langsung tersedia di SQLScript.

GROUP BY, SUM, window function, dan CASE harus dikombinasikan untuk meniru perilaku BW.

7) Kesulitan Validasi Data CDS terhadap Laporan BW

Walaupun CDS sudah menghasilkan angka, membuktikan bahwa angka tersebut benar dan sama dengan BW tidak selalu mudah, karena laporan BW sering memiliki filter dan perhitungan tersembunyi.

Di BW:

a) Filter bisa disisipkan di Query Designer.
b) Ada definisi *key figure* yang hanya aktif pada level hasil tertentu.
c) Ada *exception aggregation* dan *calculated key figure* yang tidak tercermin langsung di tabel.

Perlu dilakukans:

a) Mengeksekusi BW Query dengan filter yang sama persis dengan CDS,
b) Membandingkan angka per kombinasi dimensi (misalnya per GL, per company code, per periode),

Proses ini memerlukan iterasi berkali-kali dan diskusi dengan mentor untuk memastikan logika migrasi sudah benar.

3.3.3 Solusi atas Kendala yang Ditemukan

Menanggapi kendala-kendala yang muncul selama proses migrasi SAP BW ke ABAP CDS dan AMDP, serangkaian solusi yang disusun berpasangan dengan setiap kendala diterapkan. Dengan demikian, setiap kendala memiliki pendekatan penyelesaian yang jelas, baik dari sisi konsep maupun teknis.

1) Menyederhanakan dan Memetakan Ulang Blueprint BW

Untuk mengurangi kebingungan akibat struktur BW yang sangat kompleks dan bertingkat, langkah pertama yang dilakukan adalah "merapikan peta" BW. Artinya, alur data dari sumber hingga menjadi laporan dipecah menjadi potongan-potongan yang lebih kecil dan mudah dipahami, sebelum diterjemahkan ke CDS.

Secara teknis:

a) Mengidentifikasi InfoProvider utama dan pendukung (DSO, InfoCube, Query).
b) Hanya memilih field yang benar-benar dipakai di laporan target.
c) Mengelompokkan logic BW menjadi:
(1) *lookup master data*,
(2) *kalkulasi key figure*,
(3) *agregasi/summary*,
d) Menyusun dokumen mapping field BW → I_View/CDS, termasuk asal tabel, cara perhitungan, dan posisi field di layer CDS.

Hasil pemetaan ini menjadi referensi ketika membangun basic view dan cube view, sehingga proses migrasi tidak hanya menyalin struktur yang

rumit, tetapi mengubahnya menjadi desain data yang lebih terstruktur dan efisien.

2) Mendesain Ulang Join dengan Key yang Tepat dan Terukur

Agar penggabungan data dari berbagai tabel tidak lagi menimbulkan angka berlipat atau data hilang, join dirancang ulang dengan prinsip hanya record yang benar-benar berpasangan yang digabungkan. Kombinasi key yang digunakan tidak lagi sekadar kelihatannya sama, tetapi diuji supaya benar-benar unik.

Secara teknis:

a) Menambahkan kondisi join tambahan, misalnya valid-from/valid-to pada master.
b) Menghindari join langsung antar dua tabel yang sama-sama many-to-many tanpa perantara.
c) Memindahkan join yang kompleks ke basic view sehingga bisa dicek dan diuji lebih awal.
d) Menggunakan DISTINCT atau agregasi di titik yang tepat jika masih terdapat indikasi duplikasi.

Contoh perbaikan:

a) Join jurnal ke cost center tidak hanya berdasarkan cost center, tetapi juga periode dan company code.
b) Join material document memastikan kombinasi kunci (material, plant, movement type, posting date) sudah cukup unik.

Dengan desain join yang baru, data yang semula berlipat atau hilang dapat dinormalkan, dan hasil agregasi menjadi lebih akurat.

3) Standardisasi Tipe Data dan Penggunaan CAST pada UNION

Untuk mengatasi error UNION yang muncul karena perbedaan definisi kolom, tim sepakat menggunakan "standar tipe data" untuk field-field penting. Jika ada sumber yang berbeda, maka diseragamkan lebih dulu sebelum di-UNION.

Langkah-langkah teknis yang dilakukan:

a) Memeriksa data element/domain dari field yang akan di-UNION (misalnya amount, quantity, key karakter).
b) Memilih satu definisi "master" (misalnya amount → DEC(23,2), material → CHAR(18)).
c) Menambahkan CAST() di sisi yang tidak sesuai.
d) Memastikan urutan dan nama kolom pada kedua sisi UNION sama persis.

Dengan cara ini, error "Length of 1st occurrence must be larger" hilang, activation view menjadi lebih stabil, dan struktur field pada view yang digabung bersifat konsisten serta mudah dirawat di kemudian hari.

4) Memecah View Menjadi Beberapa Layer untuk Meningkatkan Performa

Daripada satu view menangani semua tugas (join, kalkulasi, agregasi, dan penyajian sekaligus), desain dipecah menjadi beberapa layer. Setiap layer hanya mengerjakan tugas tertentu sehingga beban kerja terbagi dan query menjadi lebih ringan.

Implementasi teknis:

a) Basic View → fokus pada join dan pemilihan field dasar yang benar,

b) Cube View → fokus pada agregasi (SUM, COUNT) dan perhitungan key figure,
c) Consumption View → fokus pada parameter, filter, dan penyajian ke aplikasi (Fiori/SAC).

Selain itu:

a) Join yang tidak dibutuhkan untuk semua laporan dipindahkan ke view tambahan (helper),
b) Perhitungan kompleks dipindahkan ke layer yang datanya sudah terfilter,
c) Annotation seperti @Analytics.dataCategory: #CUBE digunakan dengan tepat.

Dengan pendekatan ini, *query plan* SAP HANA menjadi lebih sederhana, penggunaan resource turun, dan waktu eksekusi view berkurang secara signifikan.

5) Mengurangi Ketergantungan pada AMDP dan Memodulasi Kode

Blok AMDP yang terlalu besar dan kompleks dipecah dan, sejauh mungkin, dipindahkan kembali ke CDS. Tujuannya agar logika perhitungan tersebar secara wajar dan tidak bergantung.

Beberapa langkah yang dilakukan:

a) Mengidentifikasi bagian AMDP yang hanya melakukan SELECT dan agregasi standar -> dipindah menjadi CDS biasa.
b) Menyisakan AMDP/CTF hanya untuk logika yang benar-benar membutuhkan SQLScript tingkat lanjut.

c) Merapikan AMDP menjadi beberapa bagian modular: satu AMDP untuk satu tugas logis yang spesifik, bukan satu AMDP untuk semua hal.

Dengan cara ini:

a) Proses debugging menjadi lebih mudah karena ruang lingkup tiap AMDP lebih kecil.
b) Perubahan di masa depan dapat dilakukan di layer CDS tanpa harus selalu menyunting AMDP.
c) Struktur keseluruhan arsitektur data menjadi lebih transparan dan mudah dijelaskan.

6) Menyusun Pola Mapping Function BW ke AMDP Secara Sistematis

Alih-alih menerjemahkan formula BW ke AMDP secara ad-hoc setiap kali, "kamus pola" yang memetakan jenis-jenis formula BW disusun ke pola penulisan SQLScript tertentu. Dengan begitu, proses translasi menjadi lebih konsisten dan bisa diulang.

```
 from
 (SELECT DISTINCT
     c.client
     ,c.ControllingArea
     ,c.CostCenter
     ,c.DocumentType
     ,c.Plant
     ,c.ProfitCenter
     ,c.ReferenceKey
     ,c.WBSElement
     ,case when c.SalesOffice = '' or c.SalesOffice = '0'
         THEN LTRIM(RTRIM(REPLACE_REGEXPR(' +' IN LTRIM(c.ProfitCenter,'0') with ' ')))
      ELSE
         c.SalesOffice
      end as SalesOffice
from

     (SELECT DISTINCT
         b.client
         ,b.ControllingArea
         ,b.CostCenter
         ,b.DocumentType
         ,b.Plant
         ,b.ProfitCenter
         ,b.ReferenceKey
         ,b.WBSElement
         ,case
             when ltrim(rtrim(b.ProfitCenter)) IN ('3998', LPAD('3998',10,'0')) THEN
                 CASE
                     when b.ProfitCenterConv in( '' ,'0') and substring( b.WBSElement,1,1 ) ='P' then
                         case
                             when substring( b.WBSElement,7,3 ) ='999'
                                 then '3999'
```

Gambar 3.22 Code Function Amdp di CDS

Contoh pola mapping yang digunakan:

a) IF/ELSE BW → CASE WHEN ... THEN ... ELSE ... END di SQLScript.
b) Restricted key figure → SELECT dengan kondisi WHERE yang membatasi subset data sebelum agregasi.
c) Exception aggregation (by characteristic) → kombinasi GROUP BY + window function (OVER (PARTITION BY ...)).
d) Calculated key figure → ekspresi aritmatika di SELECT.

Logika BW kompleks juga dipecah menjadi beberapa blok:

a) Blok validasi data.
b) Blok perhitungan dasar.
c) Blok agregasi dan hasil akhir.

Dengan pola ini, walaupun tidak semua function BW memiliki padanan satu-satu, perilaku akhirnya bisa ditiru dengan kombinasi SQLScript yang terstruktur.

7) Membangun Proses Validasi Data Bertahap terhadap BW

Validasi dilakukan secara bertahap, tidak langsung sampai ke level detail terkecil. Pendekatannya adalah dari global ke detail, mulai dari total besar, kemudian masuk ke rincian jika ada perbedaan. Hal ini memudahkan penelusuran sumber selisih.

Langkah validasi disusun sebagai berikut:

a) Menjalankan laporan BW dan CDS dengan filter yang sama (ledger, tahun fiskal, periode, company code).
b) Membandingkan total di level tinggi (misalnya total per company code dan tahun).
c) Jika terdapat selisih, melakukan drill-down ke level:
(1) Per GL account.
(2) Per profit center atau cost center.
(3) Per kombinasi dimensi lainnya.
d) Menggunakan data hasil drill-down untuk menelusuri:
(1) Baris yang hilang (unmatched).
(2) Baris yang berlipat (duplicate).
(3) Perbedaan filter atau agregasi.
e) Mencatat setiap kasus selisih beserta perbaikan yang dilakukan (misalnya update join, koreksi filter, atau perubahan perhitungan).

Hasil validasi ini bukan hanya memastikan bahwa CDS menghasilkan angka yang konsisten dengan BW, tetapi juga menjadi dokumentasi formal

yang menjelaskan bahwa proses migrasi telah melalui tahapan pengujian yang memadai.