BAB III

PELAKSANAAN KERJA MAGANG

3.1. Kedudukan dan Koordinasi

Di ASABA, peserta magang berkedudukan sebagai ETL (*Extract-Transform-Load*) dan *report developer* yang dikoordinasikan oleh divisi *Technology and Integration Services*. Tim *developer* untuk proyek ini berjumlah tujuh orang, termasuk peserta magang dan dikoordinasi oleh Bapak Alex Bong.

Sesuai dengan *user requirement*, tim *developer* melakukan perancangan aplikasi secara bersama. Peserta magang turut serta dalam melakukan perancangan *data modeling* dan implementasi *data warehouse*, yaitu dengan melakukan proses ETL dengan Oracle Data Integrator 11g, serta mengembangkan *report* dengan Oracle Report 11g.

3.2. Tugas yang Dilakukan

Selama periode kerja magang, peserta magang ditugasi beberapa hal, termasuk membantu melakukan *setup and configuration development environtment* pada *development server* yang disediakan oleh klien (Kementrian Keuangan). *Setup development environment* meliputi instalasi Oracle Repository Creation Utility, Oracle Database 11g, Oracle Web Logic Server, Oracle Data Integrator 11g, Oracle Development Suite 11g, Oracle Business Intelligence 11g, dan Oracle Forms and Reports 11g. Setelah setup selesai, peserta magang turut serta melakukan konfigurasi sehingga *development environment* siap pakai.

Setelah itu, peserta magang diberi tugas untuk menganalisa proses bisnis pada Kementrian Keuangan yaitu dengan menganalisa *form* dan *report* serta *package* yang sudah ada, yang kemudian dijadikan *data model.* Peserta magang kemudian mengimplementasikan *data model* tersebut menjadi *data warehouse* dengan Oracle Data Integrator 11g, dimana *data warehouse* tersebut akan menjadi *data source* untuk *report* yang akan dikembangkan. Peserta magang berperan besar dalam pembuatan *data warehouse* dan *report*, oleh karena itu kedua hal inilah yang akan diangkat sebagai pembahasan utama dalam laporan kerja magang ini.

3.3. Uraian Pelaksanaan Kerja Magang

Aplikasi Sistem Perbendaharaan dan Anggaran Negara (SPAN) menggunakan Oracle Database 11g dan Oracle E-Business Suite 11g dalam melakukan tugas perbendaharaan, penganggaran dan pelaporan keuangan negara. Pelaporan pada SPAN dilakukan dengan mengambil data secara langsung ke *transaction database*, sehingga membuat performanya menjadi kurang. Oleh karena itu, ASABA mengembangkan suatu *data warehouse* dengan konsep ETL (*Extract-Transform-Load*) yang mengambil data dari *transaction database* SPAN yang diperlukan dalam *reporting*. Setelah *data warehouse* dibuat, maka *reporting* pada SPAN akan diintegrasikan menggunakan *data warehouse*, bukan *transaction database* sehingga meningkatkan performa.

Selama periode kerja magang, peserta magang melaksanakan tugas-tugas berikut.

1. Merancang struktur *database* untuk *data warehouse*.
2. Membangun *data warehouse* sesuai dengan rancangan tersebut.
3. Mengembangkan *report* dengan menggunakan *data warehouse* yang telah dibangun sebagai *data-source*.

3.3.1. Proses Pelaksanaan Magang

Peserta magang melakukan perancangan struktur *database* untuk membangun *data warehouse* yang akan digunakan untuk melakukan *mapping* antara *data source* (tabel-tabel dari database asli) dan *data target* (tabel-tabel pada *data warehouse*). *Data source* yang dipakai dalam pembangunan *data warehouse* ini sebagai berikut.

Tabel 3.1 – Struktur Tabel FND_FLEX_VALUES_VL

No.	Nama Field	Tipe Data
1	ROW_ID	VARCHAR2 (240 Byte)
2	FLEX_VALUE_SET_ID	NUMBER
3	FLEX_VALUE_ID	NUMBER
4	FLEX_VALUE	VARCHAR2 (240 Byte)
5	LAST_UPDATE_DATE	TIMESTAMP(6)
6	LAST_UPDATED_BY	NUMBER
7	CREATION_DATE	TIMESTAMP(6)
8	CREATED_BY	NUMBER
9	LAST_UPDATE_LOGIN	NUMBER
10	ENABLED_FLAG	VARCHAR2 (240 Byte)

No.	Nama Field	Tipe Data
11	SUMMARY_FLAG	VARCHAR2 (240 Byte)
12	START_DATE_ACTIVE	TIMESTAMP(6)
13	END_DATE_ACTIVE	TIMESTAMP(6)
14	PARENT_FLEX_VALUE_LOW	VARCHAR2 (240 Byte)
15	PARENT_FLEX_VALUE_HIGH	VARCHAR2 (240 Byte)
16	STRUCTURED_HIERARCHY_LEVEL	VARCHAR2 (240 Byte)
17	HIERARCHY_LEVEL	VARCHAR2 (240 Byte)
18	COMPILED_VALUE_ATTRIBUTES	VARCHAR2 (240 Byte)
19	VALUE_CATEGORY	VARCHAR2 (240 Byte)
20	ATTRIBUTE1	VARCHAR2 (240 Byte)
21	…	VARCHAR2 (240 Byte)
69	ATTRIBUTE50	VARCHAR2 (240 Byte)
70	FLEX_VALUE_MEANING	VARCHAR2 (240 Byte)
71	DESCRIPTION	VARCHAR2 (480 Byte)
72	ATTRIBUTE_SORT_ORDER	VARCHAR2 (240 Byte)

Tabel 3.2 – Struktur Tabel FND_FLEX_VALUE_SETS

No.	Nama Field	Tipe Data
1	FLEX_VALUE_SET_ID	NUMBER (10)
2	FLEX_VALUE_SET_NAME	VARCHAR2 (60 Byte)
3	LAST_UPDATE_DATE	DATE
4	LAST_UPDATED_BY	NUMBER (15)

No.	Nama Field	Tipe Data
5	CREATION_DATE	DATE
6	CREATED_BY	NUMBER (15)
7	LAST_UPDATE_LOGIN	NUMBER (15)
8	VALIDATION_TYPE	VARCHAR2 (1 Byte)
9	PROTECTED_FLAG	VARCHAR2 (1 Byte)
10	SECURITY_ENABLED_FLAG	VARCHAR2 (1 Byte)
11	LONGLIST_FLAG	VARCHAR2 (1 Byte)
12	FORMAT_TYPE	VARCHAR2 (1 Byte)
13	MAXIMUM_SIZE	NUMBER (3)
14	ALPHANUMERIC_ALLOWED_FLAG	VARCHAR2 (1 Byte)
15	UPPERCASE_ONLY_FLAG	VARCHAR2 (1 Byte)
16	NUMERIC_MODE_ENABLED_FLAG	VARCHAR2 (1 Byte)
17	DESCRIPTION	VARCHAR2 (240 Byte)
18	DEPENDANT_DEFAULT_VALUE	VARCHAR2 (60 Byte)
19	DEPENDANT_DEFAULT_MEANING	VARCHAR2 (240 Byte)
20	PARENT_FLEX_VALUE_SET_ID	NUMBER (10)
21	MINIMUM_VALUE	VARCHAR2 (150 Byte)
22	MAXIMUM_VALUE	VARCHAR2 (150 Byte)
23	NUMBER_PRECISION	NUMBER (2)

Tabel 3.3 – Struktur Tabel GL_CODE_COMBINATIONS

No.	Nama Field	Tipe Data

No.	Nama Field	Tipe Data
1	CODE_COMBINATION_ID	NUMBER (15)
2	LAST_UPDATE_DATE	DATE
3	LAST_UPDATED_BY	NUMBER
4	CHART_OF_ACCOUNTS_ID	NUMBER (15)
5	DETAIL_POSTING_ALLOWED_FLAG	VARCHAR2 (1 Byte)
6	DETAIL_BUDGETING_ALLOWED_FLAG	VARCHAR2 (1 Byte)
7	ACCOUNT_TYPE	VARCHAR2 (1 Byte)
8	ENABLED_FLAG	VARCHAR2 (1 Byte)
9	SUMMARY_FLAG	VARCHAR2 (1 Byte)
10	SEGMENT1	VARCHAR2 (25 Byte)
11	…	VARCHAR2 (25 Byte)
39	SEGMENT30	VARCHAR2 (25 Byte)
40	DESCRIPTION	VARCHAR2 (240 Byte)
41	TEMPLATE_ID	NUMBER (15)
42	ALLOCATION_CREATE_FLAG	VARCHAR2 (1 Byte)
43	START_DATE_ACTIVE	DATE
44	END_DATE_ACTIVE	DATE
45	ATTRIBUTE1	VARCHAR2 (150 Byte)
46	ATTRIBUTE2	VARCHAR2 (150 Byte)
47	ATTRIBUTE3	VARCHAR2 (150 Byte)
48	ATTRIBUTE4	VARCHAR2 (150 Byte)

No.	Nama Field	Tipe Data
49	ATTRIBUTE5	VARCHAR2 (150 Byte)
50	ATTRIBUTE6	VARCHAR2 (150 Byte)
51	ATTRIBUTE7	VARCHAR2 (150 Byte)
52	ATTRIBUTE8	VARCHAR2 (150 Byte)
53	ATTRIBUTE9	VARCHAR2 (150 Byte)
54	ATTRIBUTE10	VARCHAR2 (150 Byte)
55	CONTEXT	VARCHAR2 (150 Byte)
56	SEGMENT_ATTRIBUTE1	VARCHAR2 (60 Byte)
57	…	VARCHAR2 (60 Byte)
97	SEGMENT_ATTRIBUTE42	VARCHAR2 (60 Byte)
98	REFERENCE1	VARCHAR2 (1 Byte)
99	REFERENCE2	VARCHAR2 (1 Byte)
100	REFERENCE3	VARCHAR2 (25 Byte)
101	REFERENCE4	VARCHAR2 (1 Byte)
102	REFERENCE5	VARCHAR2 (1 Byte)
103	JGZZ_RECON_FLAG	VARCHAR2 (1 Byte)
104	JGZZ_RECON_CONTEXT	VARCHAR2 (30 Byte)
105	PRESERVE_FLAG	VARCHAR2 (1 Byte)
106	REFRESH_FLAG	VARCHAR2 (1 Byte)
107	IGI_BALANCED_BUDGET_FLAG	VARCHAR2 (1 Byte)
108	COMPANY_COST_CENTER_ORG_ID	NUMBER (15)

No.	Nama Field	Tipe Data
109	REVALUATION_ID	NUMBER (15)
110	LEDGER_SEGMENT	VARCHAR2 (20 Byte)
111	LEDGER_TYPE_CODE	VARCHAR2 (1 Byte)
112	ALTERNATE_CODE_COMBINATION_ID	NUMBER (15)

Tabel 3.4 – Struktur Tabel RG_REPORT_AXIS_SETS

No.	Nama Field	Tipe Data
1	APPLICATION_ID	NUMBER (15)
2	AXIS_SET_ID	NUMBER (15)
3	LAST_UPDATE_DATE	DATE
4	LAST_UPDATED_BY	NUMBER (15)
5	LAST_UPDATE_LOGIN	NUMBER (15)
6	CREATION_DATE	DATE
7	CREATED_BY	NUMBER (15)
8	NAME	VARCHAR2 (30 Byte)
9	AXIS_SET_TYPE	VARCHAR2 (1 Byte)
10	DISPLAY_IN_LIST_FLAG	VARCHAR2 (1 Byte)
11	PERIOD_SET_NAME	VARCHAR2 (15 Byte)
12	DESCRIPTION	VARCHAR2 (240 Byte)
13	ROW_SET_TITLE	VARCHAR2 (240 Byte)
14	SEGMENT_NAME	VARCHAR2 (30 Byte)
15	ID_FLEX_CODE_11I	VARCHAR2 (4 Byte)

No.	Nama Field	Tipe Data
16	STRUCTURE_ID	NUMBER (15)
17	CONTEXT	VARCHAR2 (30 Byte)
18	ATTRIBUTE1	VARCHAR2 (150 Byte)
19	…	VARCHAR2 (150 Byte)
32	ATTRIBUTE15	VARCHAR2 (150 Byte)
33	COLUMN_SET_HEADER	LONG
34	TAXONOMY_ID	NUMBER (15)
35	ID_FLEX_CODE	VARCHAR2 (4 Byte)
36	SECURITY_FLAG	VARCHAR2 (1 Byte)
37	SEEDED_NAME	VARCHAR2 (30 Byte)

Tabel 3.5 – Struktur Tabel RG_REPORT_AXES

No.	Nama Field	Tipe Data
1	APPLICATION_ID	NUMBER (15)
2	AXIS_SET_ID	NUMBER (15)
3	AXIS_SEQ	NUMBER
4	LAST_UPDATE_DATE	DATE
5	LAST_UPDATED_BY	NUMBER (15)
6	LAST_UPDATE_LOGIN	NUMBER (15)
7	CREATION_DATE	DATE
8	CREATED_BY	NUMBER (15)
9	AXIS_TYPE	VARCHAR2 (1 Byte)

No.	Nama Field	Tipe Data
10	AXIS_NAME	VARCHAR2 (30 Byte)
11	AMOUNT_ID	NUMBER (15)
12	STANDARD_AXIS_ID	NUMBER (15)
13	WIDTH	NUMBER (15)
14	POSITION	NUMBER (15)
15	STRUCTURE_ID	NUMBER (15)
16	UNIT_OF_MEASURE_ID	VARCHAR2 (30 Byte)
17	PARAMETER_NUM	NUMBER (15)
18	PERIOD_OFFSET	NUMBER (15)
19	DESCRIPTION	VARCHAR2 (240 Byte)
20	DISPLAY_FLAG	VARCHAR2 (1 Byte)
21	BEFORE_AXIS_STRING	VARCHAR2 (10 Byte)
22	AFTER_AXIS_STRING	VARCHAR2 (10 Byte)
23	NUMBER_CHARACTERS_INDENTED	NUMBER (15)
24	PAGE_BREAK_AFTER_FLAG	VARCHAR2 (1 Byte)
25	PAGE_BREAK_BEFORE_FLAG	VARCHAR2 (1 Byte)
26	NUMBER_LINES_SKIPPED_BEFORE	NUMBER (15)
27	NUMBER_LINES_SKIPPED_AFTER	NUMBER (15)
28	DISPLAY_LEVEL	NUMBER (15)
29	DISPLAY_ZERO_AMOUNT_FLAG	VARCHAR2 (1 Byte)
30	CHANGE_SIGN_FLAG	VARCHAR2 (1 Byte)

No.	Nama Field	Tipe Data
31	CHANGE_VARIANCE_SIGN_FLAG	VARCHAR2 (1 Byte)
32	DISPLAY_UNITS	NUMBER (15)
33	DISPLAY_FORMAT	VARCHAR2 (30 Byte)
34	CALCULATION_PRECEDENCE_FLAG	VARCHAR2 (1 Byte)
35	CONTEXT	VARCHAR2 (30 Byte)
36	ATTRIBUTE1	VARCHAR2 (150 Byte)
37	…	VARCHAR2 (150 Byte)
50	ATTRIBUTE15	VARCHAR2 (150 Byte)
51	PERCENTAGE_DIVISOR_SEQ	NUMBER
52	TRANSACTION_FLAG	VARCHAR2 (1 Byte)
53	FORMAT_BEFORE_TEXT	VARCHAR2 (30 Byte)
54	FORMAT_AFTER_TEXT	VARCHAR2 (30 Byte)
55	FORMAT_MASK_WIDTH	NUMBER (15)
56	DISPLAY_PRECISION	NUMBER (15)
57	SEGMENT_OVERRIDE_VALUE	VARCHAR2 (60 Byte)
58	ELEMENT_ID	NUMBER (15)
59	OVERRIDE_ALC_LEDGER_CURRENCY	VARCHAR2 (30 Byte)

Tabel 3.6 – Struktur Tabel RG_REPORT_AXIS_CONTENTS

No.	Nama Field	Tipe Data
1	APPLICATION_ID	NUMBER (15)
2	AXIS_SET_ID	NUMBER (15)

No.	Nama Field	Tipe Data
3	AXIS_SEQ	NUMBER
4	LAST_UPDATE_DATE	DATE
5	LAST_UPDATED_BY	NUMBER (15)
6	LAST_UPDATE_LOGIN	NUMBER (15)
7	CREATION_DATE	DATE
8	CREATED_BY	NUMBER (15)
9	RANGE_MODE	VARCHAR2 (1 Byte)
10	SIGN	VARCHAR2 (1 Byte)
11	DR_CR_NET_CODE	VARCHAR2 (1 Byte)
12	LEDGER_ID	NUMBER (15)
13	SEGMENT1_LOW	VARCHAR2 (60 Byte)
14	SEGMENT1_HIGH	VARCHAR2 (60 Byte)
15	SEGMENT1_TYPE	VARCHAR2 (1 Byte)
16	…	VARCHAR2 (60 Byte)
17	…	VARCHAR2 (60 Byte)
18	…	VARCHAR2 (1 Byte)
100	SEGMENT30_LOW	VARCHAR2 (60 Byte)
101	SEGMENT30_HIGH	VARCHAR2 (60 Byte)
102	SEGMENT30_TYPE	VARCHAR2 (1 Byte)
103	CONTEXT	VARCHAR2 (30 Byte)
104	ATTRIBUTE1	VARCHAR2 (150 Byte)

No.	Nama Field	Tipe Data
105	…	VARCHAR2 (150 Byte)
118	ATTRIBUTE15	VARCHAR2 (150 Byte)
119	SEGMENT_ATTRIBUTE1_LOW	VARCHAR2 (60 Byte)
120	SEGMENT_ATTRIBUTE1_HIGH	VARCHAR2 (60 Byte)
121	SEGMENT_ATTRIBUTE1_TYPE	VARCHAR2 (1 Byte)
122	…	VARCHAR2 (60 Byte)
123	…	VARCHAR2 (60 Byte)
124	…	VARCHAR2 (1 Byte)
242	SEGMENT_ATTRIBUTE42_LOW	VARCHAR2 (60 Byte)
243	SEGMENT_ATTRIBUTE42_HIGH	VARCHAR2 (60 Byte)
244	SEGMENT_ATTRIBUTE42_TYPE	VARCHAR2 (1 Byte)
245	LEDGER_SEGMENT_TYPE	VARCHAR2 (1 Byte)
246	ALC_LEDGER_CURRENCY	VARCHAR2 (30 Byte)

Mapping antara *data source* dan *data target* dilakukan dengan menggunakan Oracle Data Integrator 11g yaitu dengan proses *Extract-Transform-Load* (ETL) sebagai berikut.

1. Proses ETL pada DIM_INTRACO

Tabel 3.7 – Struktur Tabel DIM_INTRACO

No.	Nama Field	Tipe Data	Keterangan
1	INTRACO_ID	NUMBER	*Surrogate Key (PK)*
2	KODE_INTRACO	VARCHAR2 (10 Byte)	*Natural Key*

No.	Nama Field	Tipe Data	Keterangan
3	NAMA_INTRACO	VARCHAR2 (240 Byte)	*Add Row on Change*
4	CUR_IDX	NUMBER	*Current Record Flag*
5	START_DATE	DATE	*Starting Timestamp*
6	END_DATE	DATE	*Ending Timestamp*

Query yang digunakan untuk meng-*extract* data dari *data source*, meng-*transform* lalu me-*load*-nya ke *data warehouse*.

```
SELECT ffv.flex_value  AS kode_intraco,
       ffv.description AS nama_intraco
FROM   gl_code_combinations gcc,
       fnd_flex_values_vl ffv
WHERE  gcc.segment11=ffv.flex_value;
```

Gambar 3.1 – *Query* untuk DIM_SATKER

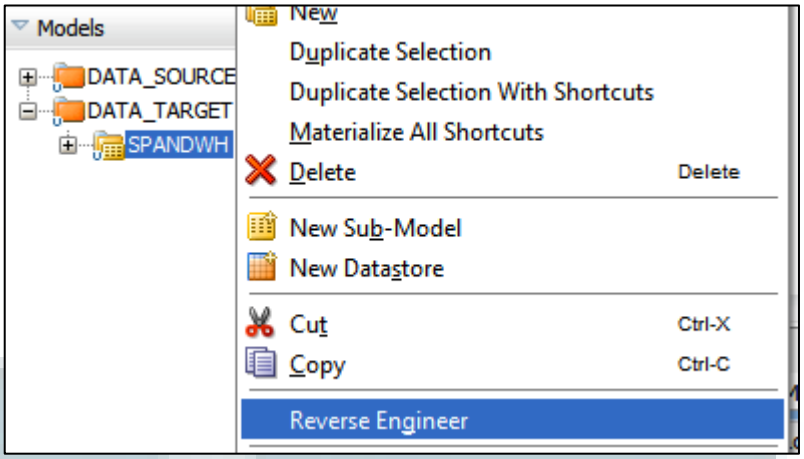
Setelah tabel target dan *query* disiapkan, dilakukan proses ETL dengan Oracle Data Integrator 11g.

a. Membuat *sequence* sebagai *Surrogate Key* pada *data warehouse*.

```
CREATE SEQUENCE DIM_INTRACO_SEQ
START WITH 1 MINVALUE 1 NOMAXVALUE;
```

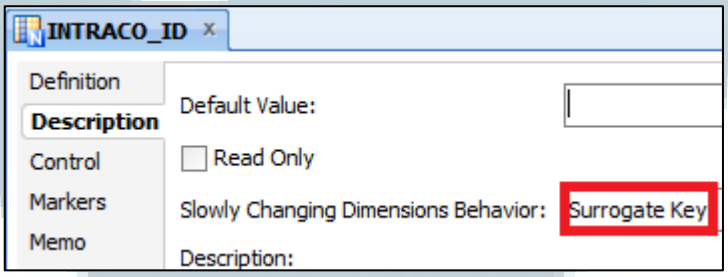
Gambar 3.2 – DDL untuk membuat DIM_INTRACO_SEQ

b. Melakukan *reverse engineer* pada Oracle Data Integrator.

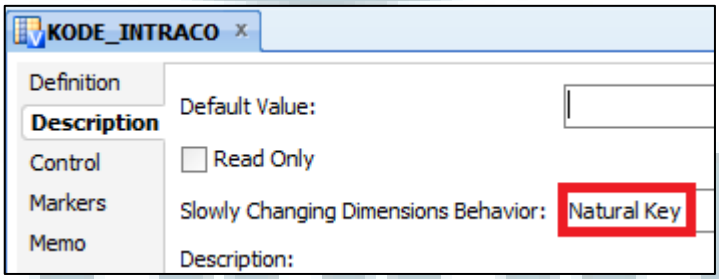


Gambar 3.3 – *Reverse Engineering* untuk DIM_INTRACO

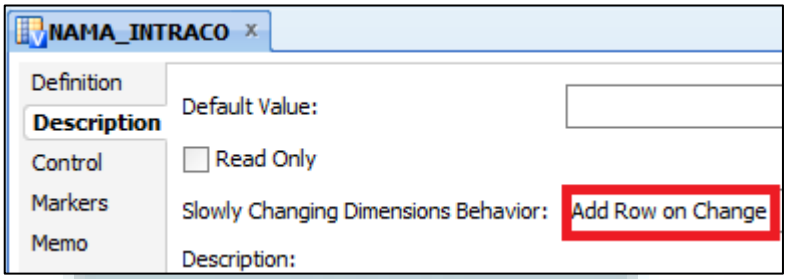
c. Mengidentifikasikan tipe *Slowly Changing Dimensions* (SCD) *Behavior* untuk setiap kolom pada tabel target.



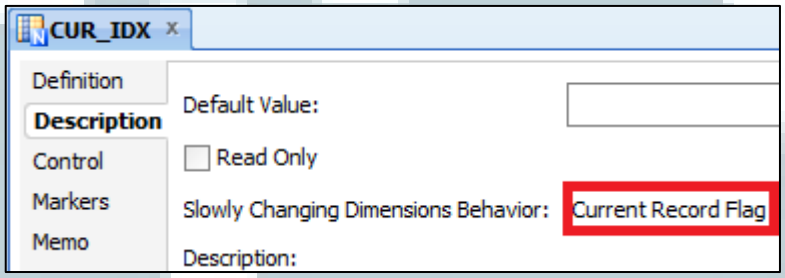
Gambar 3.4 – Spesifikasi *behavior* kolom INTRACO_ID



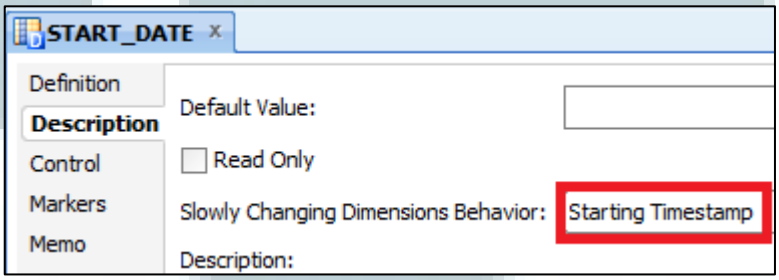
Gambar 3.5 – Spesifikasi *behavior* kolom KODE_INTRACO



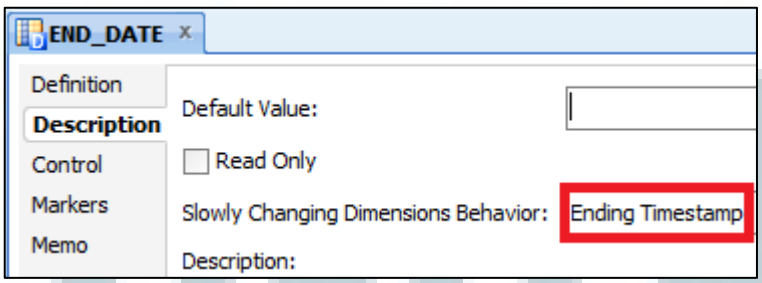
Gambar 3.6 – Spesifikasi *behavior* kolom NAMA_INTRACO



Gambar 3.7 – Spesifikasi *behavior* kolom CUR_IDX

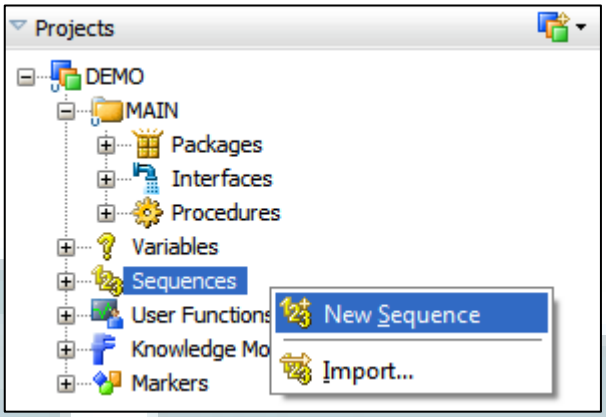


Gambar 3.8 – Spesifikasi *behavior* kolom START_DATE

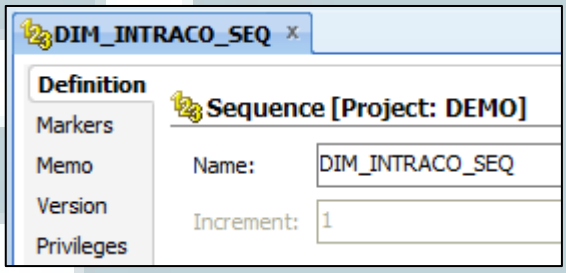


Gambar 3.9 – Spesifikasi *behavior* kolom END_DATE

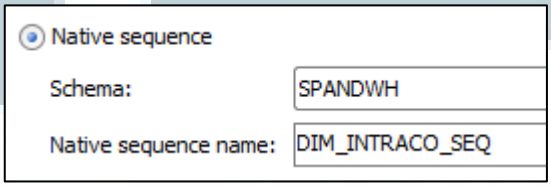
d. Membuat *project sequence* pada Oracle Data Integrator.



Gambar 3.10 – Membuat DIM_INTRACO_SEQ pada ODI(Langkah 1)

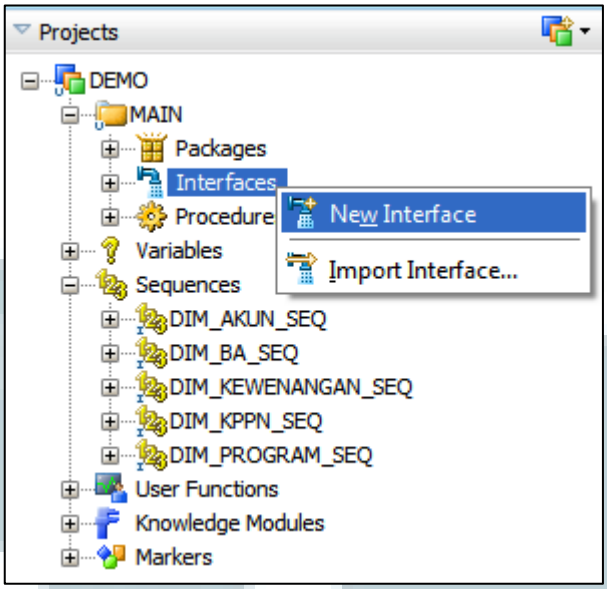


Gambar 3.11 – Membuat DIM_INTRACO_SEQ pada ODI(Langkah 2)

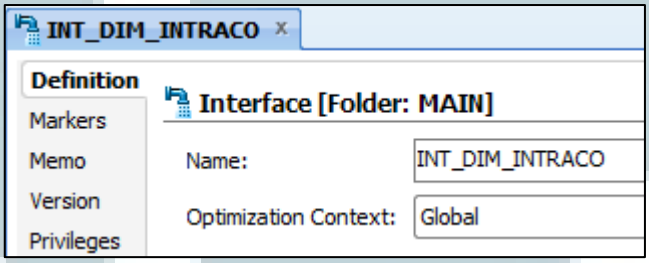


Gambar 3.12 – Membuat DIM_INTRACO_SEQ pada ODI(Langkah 3)

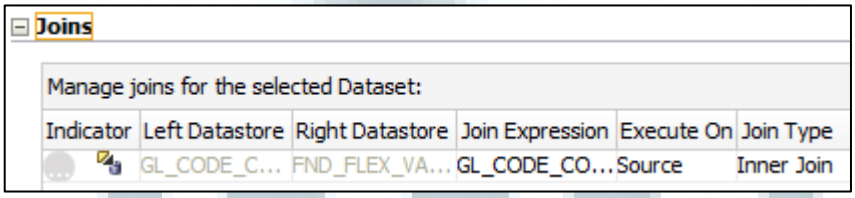
e. Melakukan *mapping* antara *data source* dan *data target* yaitu dengan membuat *interface* pada Oracle Data Integrator dimana dilakukan pengidentifikasian hubungan antara *data source* dengan *data target* yang ingin diambil.



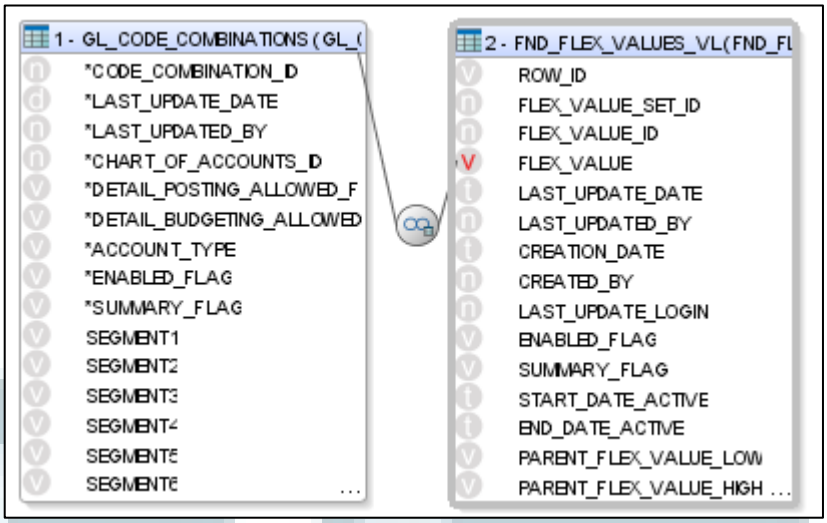
Gambar 3.13 – Membuat INT_DIM_INTRACO (Langkah 1)



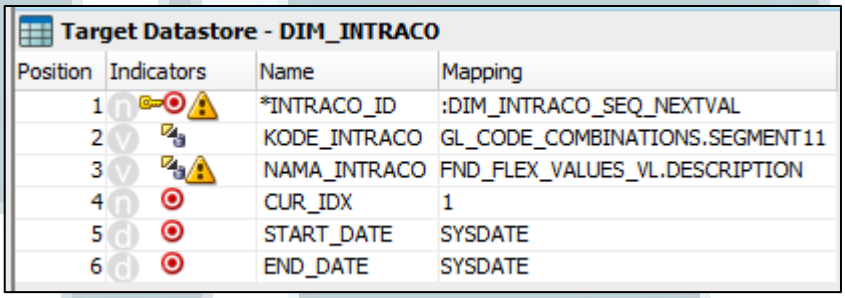
Gambar 3.14 – Membuat INT_DIM_INTRACO (Langkah 2)



Gambar 3.15 – Membuat INT_DIM_INTRACO (Langkah 3)

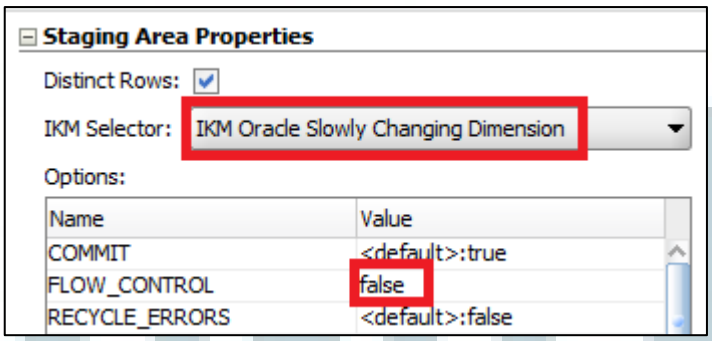


Gambar 3.16 – Membuat INT_DIM_INTRACO (Langkah 4)



Gambar 3.17 – *Mapping* antara *data source* dengan DIM_INTRACO

f. Melakukan pengaturan *staging area*.



Gambar 3.18 – Mengatur *knowledge module* dan *flow control*

g. Menjalankan (*run*) *interface* yang telah dibuat, sehingga proses ETL akan berjalan dan data akan masuk dari *data source* ke *data target*.

Gambar 3.19 – Menjalankan INT_DIM_INTRACO

2. Proses ETL pada DIM_BA

Tabel 3.8 – Struktur Tabel DIM_BA

No.	Nama Field	Tipe Data	Keterangan
1	BA_ID	NUMBER	*Surrogate Key (PK)*
2	KODE_BA	VARCHAR2 (10 Byte)	*Natural Key*
3	NAMA_BA	VARCHAR2 (240 Byte)	*Add Row on Change*
4	CUR_IDX	NUMBER	*Current Record Flag*
5	START_DATE	DATE	*Starting Timestamp*
6	END_DATE	DATE	*Ending Timestamp*

Query yang digunakan untuk meng-*extract* data dari *data source*, meng-*transform* lalu me-*load*-nya ke *data warehouse*.

```
SELECT ffv.flex_value  AS kode_ba,
       ffv.description AS nama_ba
FROM   fnd_flex_value_sets ffvs,
       fnd_flex_values_vl ffv
WHERE  ffvs.flex_value_set_id=ffv.flex_value_set_id
AND    ffvs.flex_value_set_name='SPAN_DFF_BA';
```

Gambar 3.20 – *Query* untuk DIM_BA

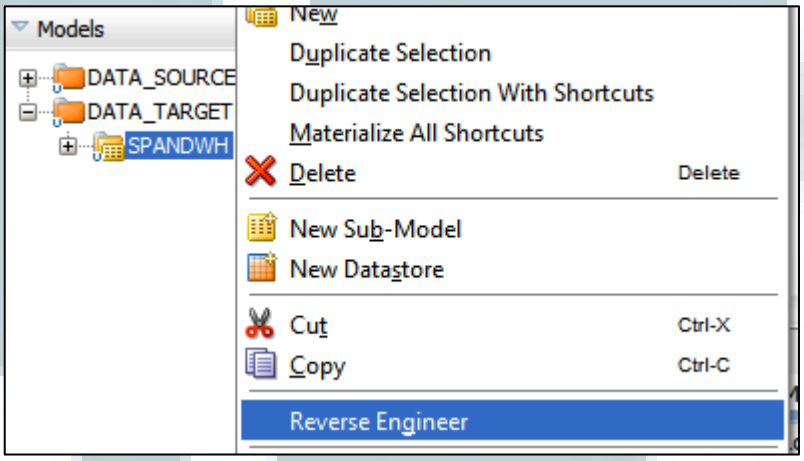
Setelah tabel target dan *query* disiapkan, dilakukan proses ETL dengan Oracle Data Integrator 11g.

a. Membuat *sequence* sebagai *Surrogate Key* pada *data warehouse.*

```
CREATE SEQUENCE DIM_BA_SEQ
START WITH 1 MINVALUE 1 NOMAXVALUE;
```

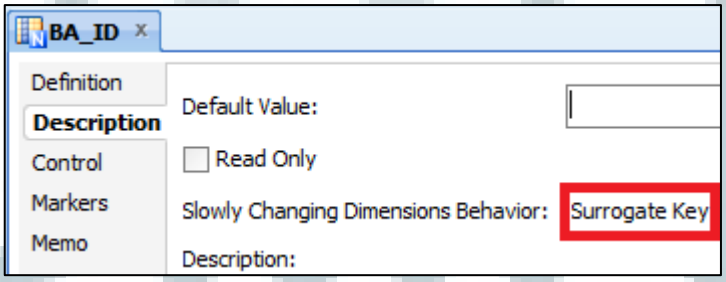
Gambar 3.21 – DDL untuk DIM_BA_SEQ

b. Melakukan *reverse engineer* pada Oracle Data Integrator.

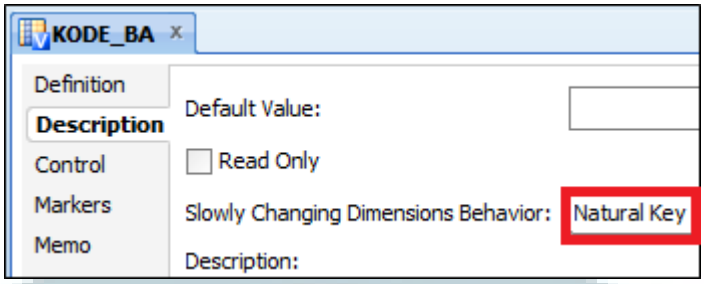


Gambar 3.22 – *Reverse Engineer* untuk DIM_BA

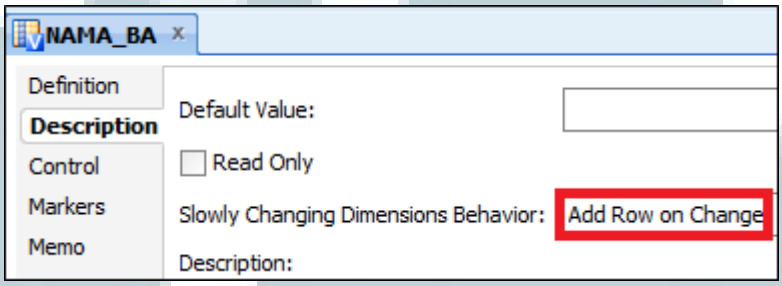
c. Mengidentifikasikan tipe *Slowly Changing Dimensions* (SCD) *Behavior* untuk setiap kolom pada tabel target.



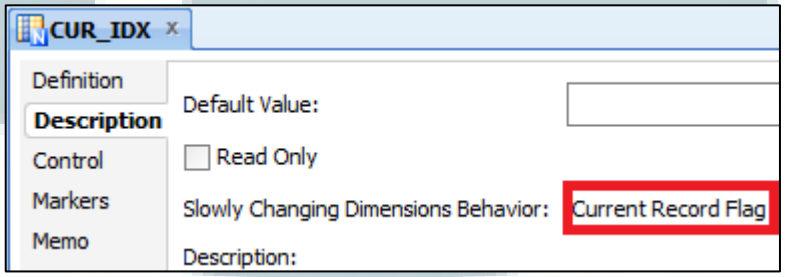
Gambar 3.23 – Spesifikasi *behavior* kolom BA_ID



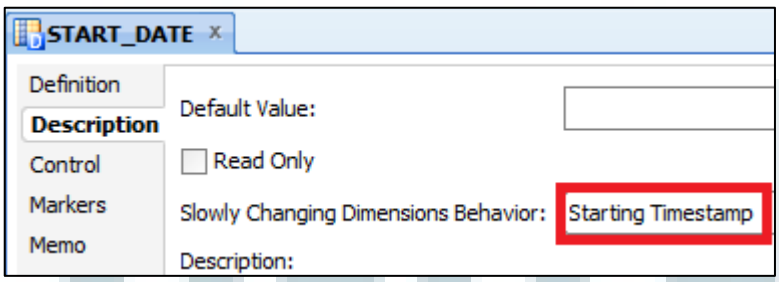
Gambar 3.24 – Spesifikasi *behavior* kolom KODE_BA



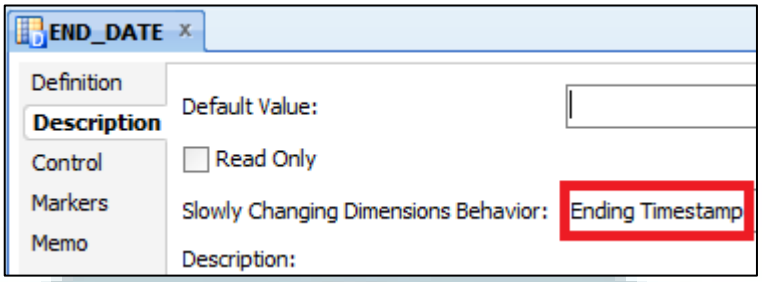
Gambar 3.25 – Spesifikasi *behavior* kolom NAMA_BA



Gambar 3.26 – Spesifikasi *behavior* kolom CUR_IDX

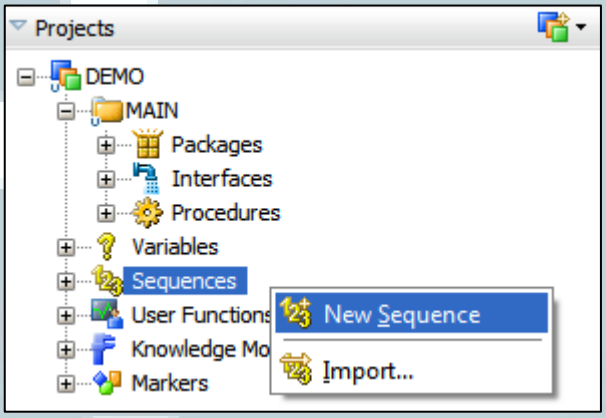


Gambar 3.27 – Spesifikasi *behavior* kolom START_DATE

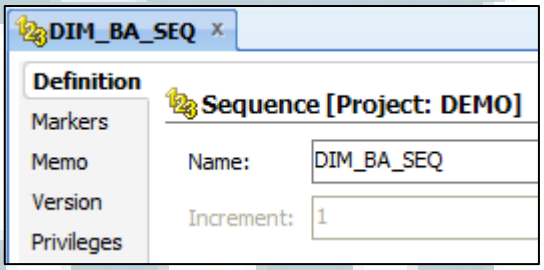


Gambar 3.28 – Spesifkasi *behavior* kolom END_DATE

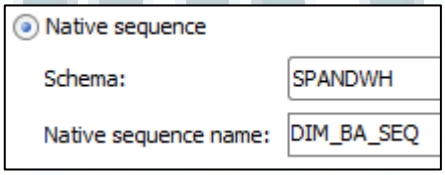
d. Membuat *project sequence* pada Oracle Data Integrator.



Gambar 3.29 – Membuat DIM_BA_SEQ pada ODI (Langkah 1)



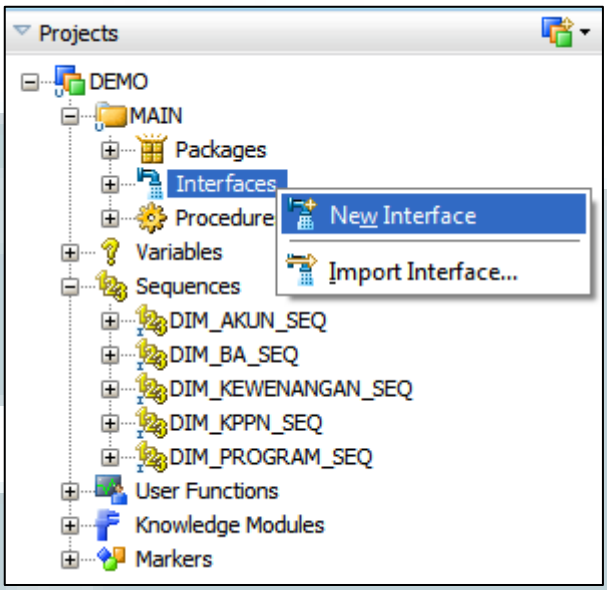
Gambar 3.30 – Membuat DIM_BA_SEQ pada ODI (Langkah 2)



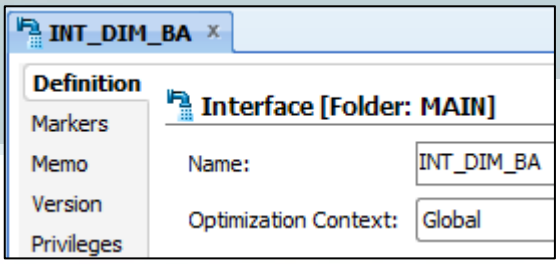
Gambar 3.31 – Membuat DIM_BA_SEQ pada ODI (Langkah 3)

e. Melakukan *mapping* antara *data source* dan *data target* yaitu dengan membuat *interface* pada Oracle Data Integrator dimana dilakukan

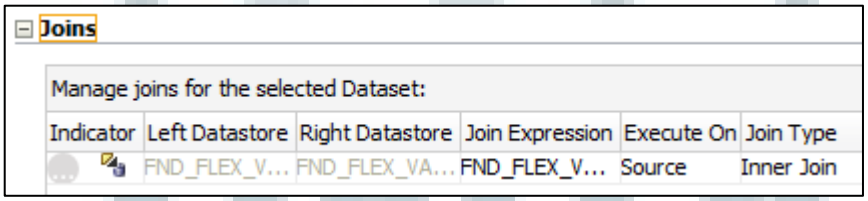
pengidentifikasian hubungan antara *data source* dengan *data target* yang ingin diambil.



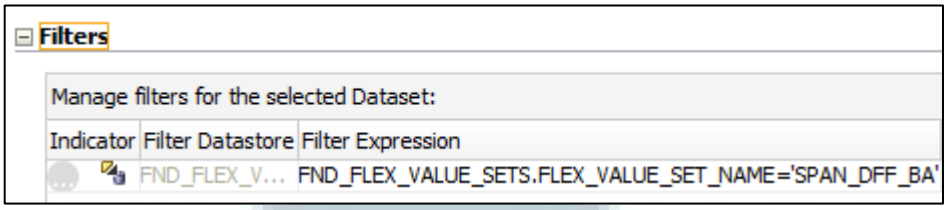
Gambar 3.32 – Membuat INT_DIM_BA (Langkah 1)



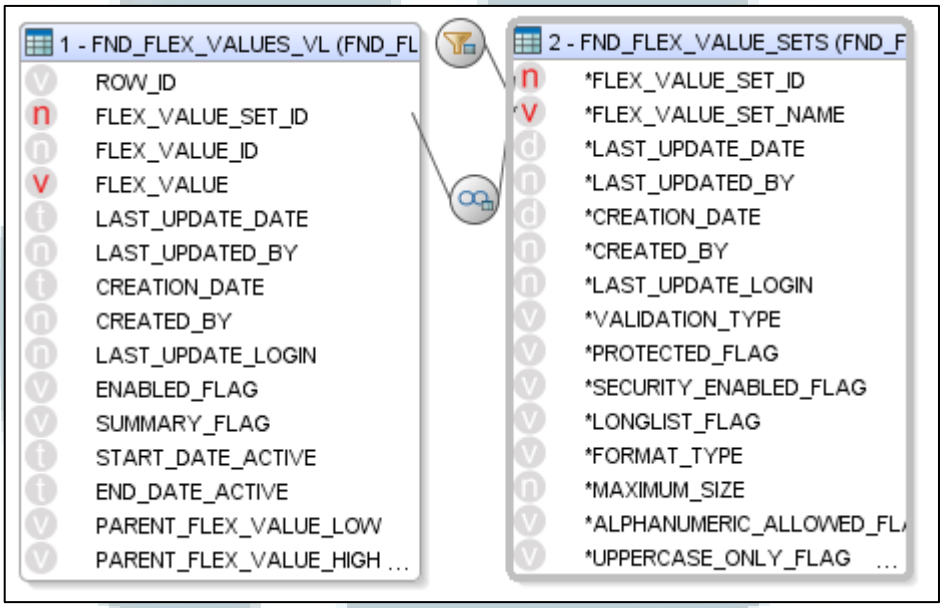
Gambar 3.33 – Membuat INT_DIM_BA (Langkah 2)



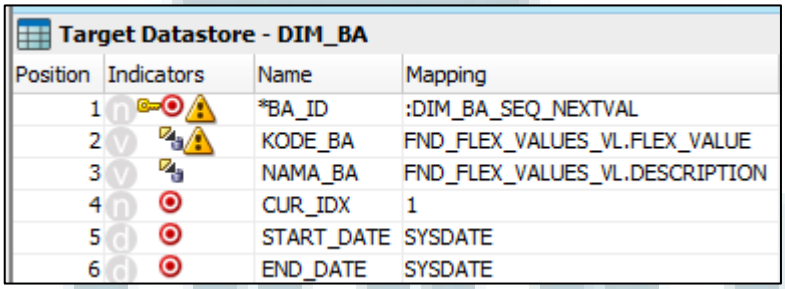
Gambar 3.34 – Membuat INT_DIM_BA (Langkah 3)



Gambar 3.35 – Membuat INT_DIM_BA (Langkah 4)

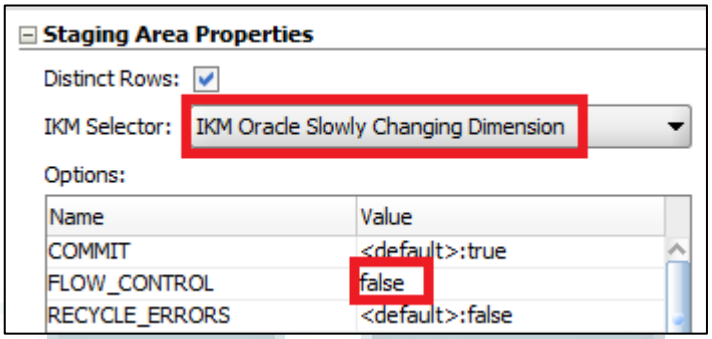


Gambar 3.36 – Membuat INT_DIM_BA (Langkah 5)



Gambar 3.37 – *Mapping* antara *data source* dan DIM_BA

f. Melakukan pengaturan *staging area*.



Gambar 3.38 – Mengatur *knowledge module* dan *flow control*

g. Menjalankan (*run*) *interface* yang telah dibuat, sehingga proses ETL akan berjalan dan data akan masuk dari *data source* ke *data target*.

Gambar 3.39 – Menjalankan INT_DIM_BA

3. Proses ETL pada DIM_KANWIL

Tabel 3.9 – Struktur Tabel DIM_KANWIL

No.	Nama Field	Tipe Data	Keterangan
1	KANWIL_ID	NUMBER	*Surrogate Key (PK)*
2	KODE_KANWIL	VARCHAR2 (10 Byte)	*Natural Key*
3	NAMA_KANWIL	VARCHAR2 (240 Byte)	*Add Row on Change*
4	CUR_IDX	NUMBER	*Current Record Flag*
5	START_DATE	DATE	*Starting Timestamp*

No.	Nama Field	Tipe Data	Keterangan
6	END_DATE	DATE	*Ending Timestamp*

Query yang digunakan untuk meng-*extract* data dari *data source*, meng-*transform* lalu me-*load*-nya ke *data warehouse.*

```
SELECT ffv.flex_value  AS kode_kanwil,
       ffv.description AS nama_kanwil
FROM   fnd_flex_value_sets ffvs,
       fnd_flex_values_vl ffv
WHERE  ffvs.flex_value_set_id=ffv.flex_value_set_id
AND    ffvs.flex_value_set_name='SPAN_DFF_KANWIL_KL_KEMKEU';
```

Gambar 3.40 – *Query* untuk DIM_KANWIL

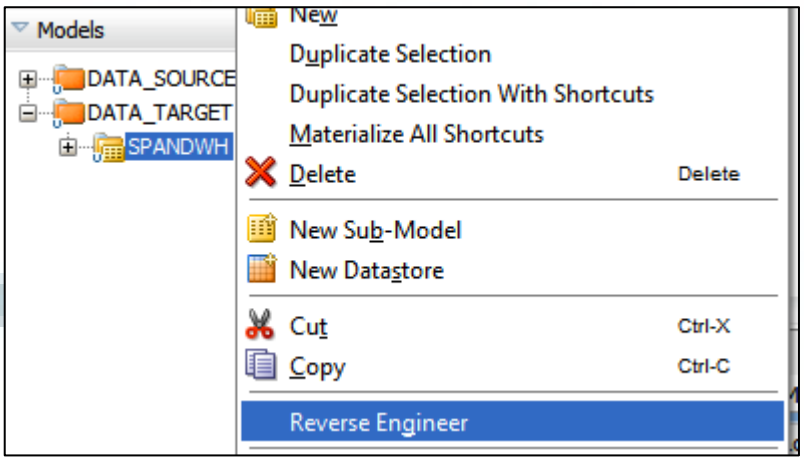
Setelah tabel target dan *query* disiapkan, dilakukan proses ETL dengan Oracle Data Integrator 11g.

a. Membuat *sequence* sebagai *Surrogate Key* pada *data warehouse.*

```
CREATE SEQUENCE DIM_KANWIL_SEQ
START WITH 1 MINVALUE 1 NOMAXVALUE;
```

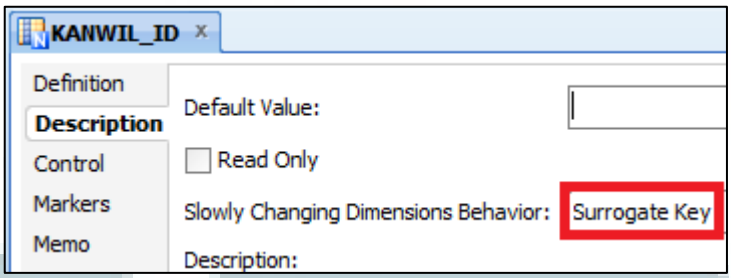
Gambar 3.41 – DDL untuk DIM_KANWIL_SEQ

b. Melakukan *reverse engineer* pada Oracle Data Integrator.

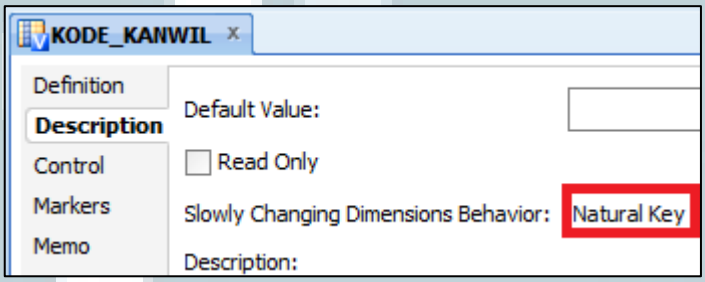


Gambar 3.42 – *Reverse Engineer* untuk DIM_KANWIL

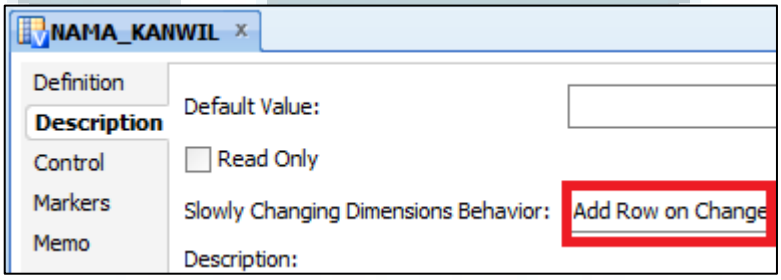
c. Mengidentifikasikan tipe *Slowly Changing Dimensions* (SCD) *Behavior* untuk setiap kolom pada tabel target.



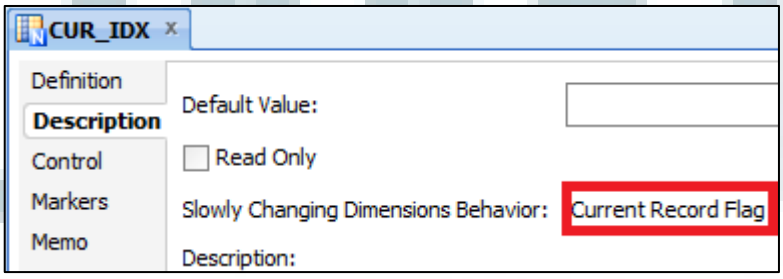
Gambar 3.43 – Spesifikasi *behavior* kolom KANWIL_ID



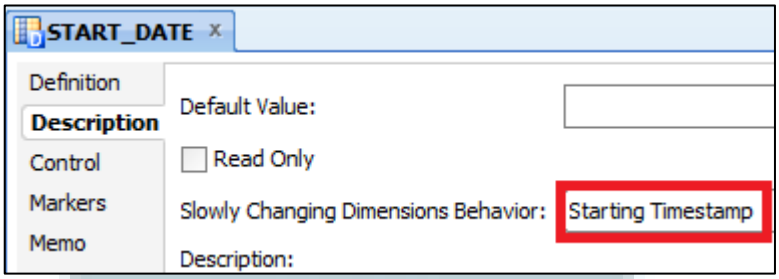
Gambar 3.44 – Spesifikasi *behavior* kolom KODE_KANWIL



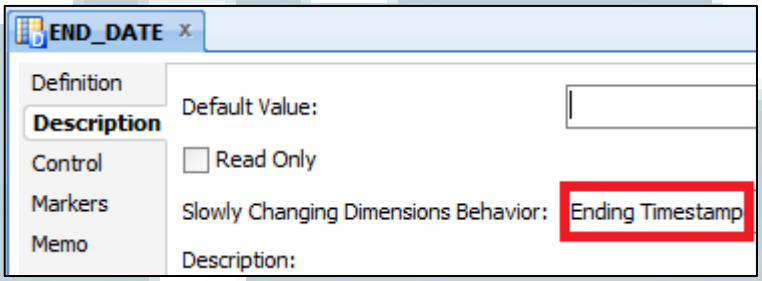
Gambar 3.45 – Spesifikasi *behavior* kolom NAMA_KANWIL



Gambar 3.46 – Spesifikasi *behavior* kolom CUR_IDX

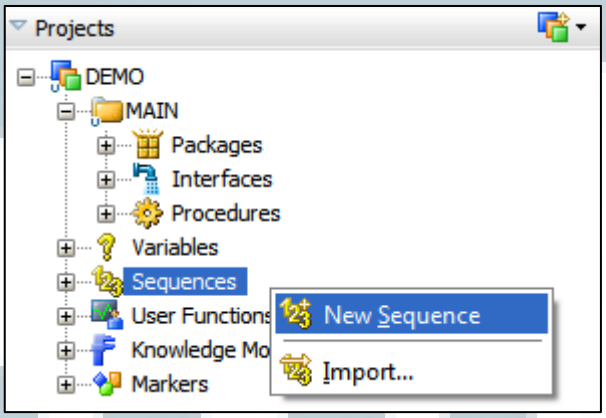


Gambar 3.47 – Spesifikasi *behavior* kolom START_DATE

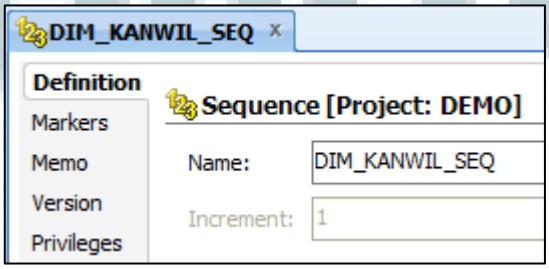


Gambar 3.48 – Spesifikasi *behavior* kolom END_DATE

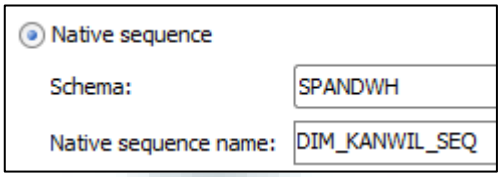
d. Membuat *project sequence* pada Oracle Data Integrator.



Gambar 3.49 – Membuat DIM_KANWIL_SEQ pada ODI (Langkah 1)

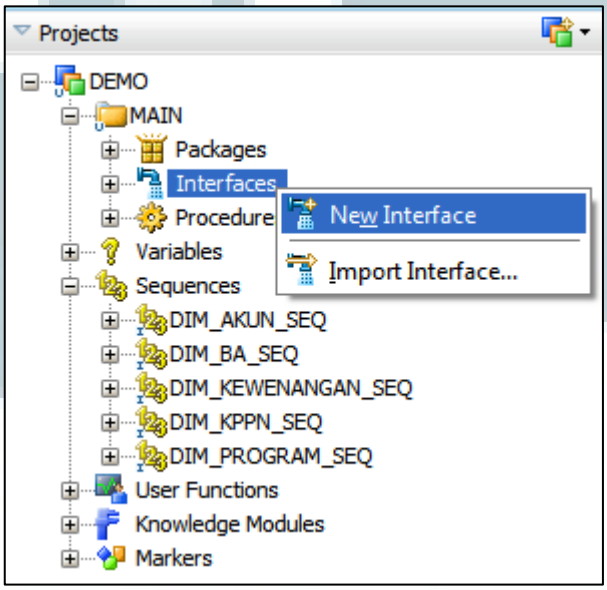


Gambar 3.50 – Membuat DIM_KANWIL_SEQ pada ODI (Langkah 2)

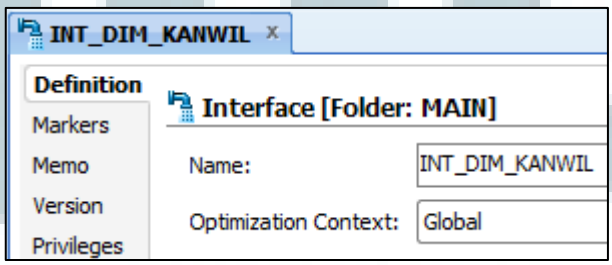


Gambar 3.51 – Membuat DIM_KANWIL_SEQ pada ODI (Langkah 3)

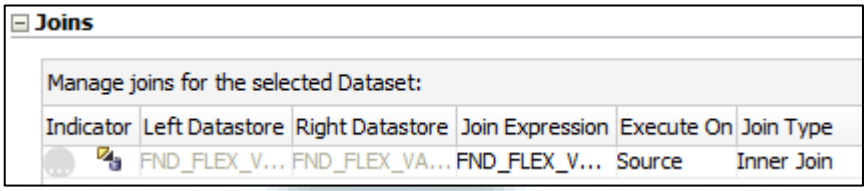
e. Melakukan *mapping* antara *data source* dan *data target* yaitu dengan membuat *interface* pada Oracle Data Integrator dimana dilakukan pengidentifikasian hubungan antara *data source* dengan *data target* yang ingin diambil.



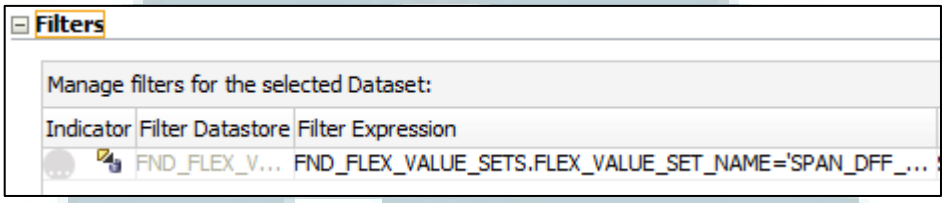
Gambar 3.52 – Membuat INT_DIM_KANWIL (Langkah 1)



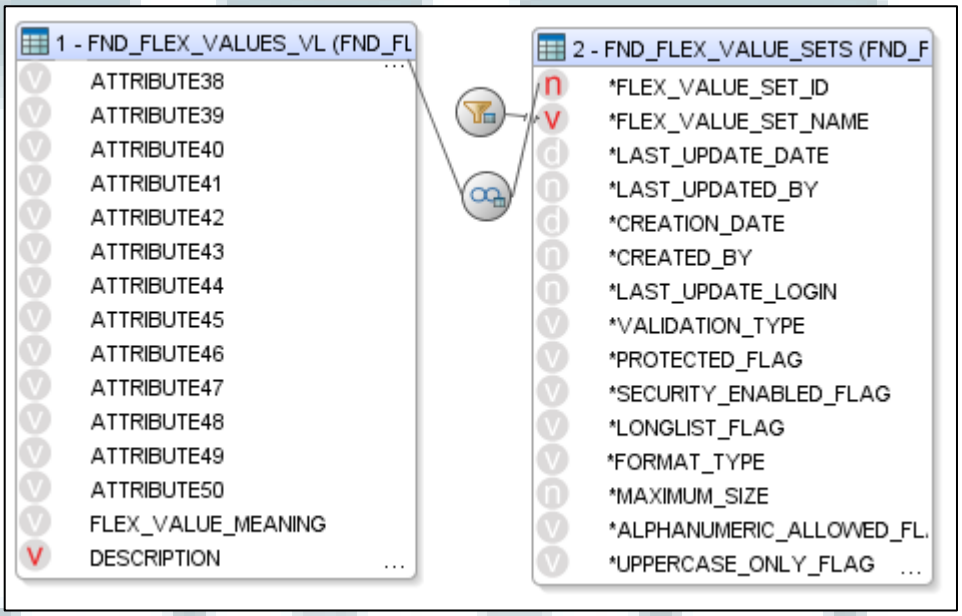
Gambar 3.53 – Membuat INT_DIM_KANWIL (Langkah 2)



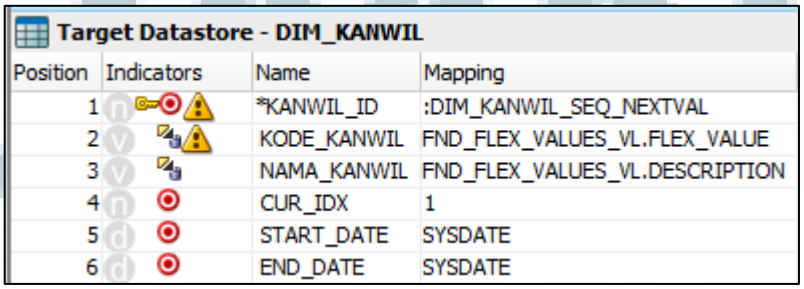
Gambar 3.54 – Membuat INT_DIM_KANWIL (Langkah 3)



Gambar 3.55 – Membuat INT_DIM_KANWIL (Langkah 4)

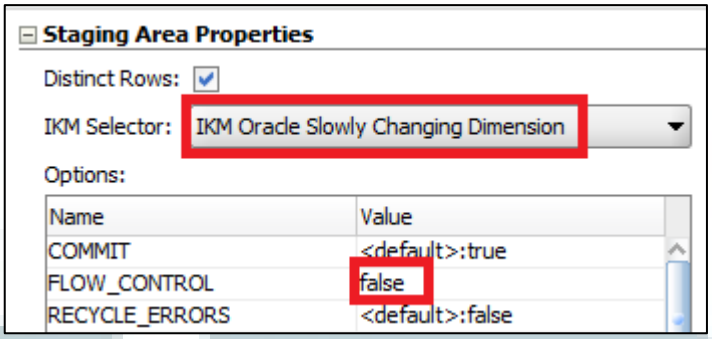


Gambar 3.56 – Membuat INT_DIM_KANWIL (Langkah 5)



Gambar 3.57 – *Mapping* antara *data source* dan DIM_KANWIL

f. Melakukan pengaturan *staging area*.



Gambar 3.58 – Mengatur *knowledge module* dan *flow control*

g. Menjalankan (*run*) *interface* yang telah dibuat, sehingga proses ETL akan berjalan dan data akan masuk dari *data source* ke *data target*.

Gambar 3.59 – Menjalankan INT_DIM_KANWIL

4. Proses ETL pada DIM_ESELON

Tabel 3.10 – Struktur Tabel DIM_ESELON

No.	Nama Field	Tipe Data	Keterangan
1	ESELON_ID	NUMBER	*Surrogate Key (PK)*
2	KODE_ESELON	VARCHAR2 (10 Byte)	*Natural Key*
3	NAMA_ESELON	VARCHAR2 (240 Byte)	*Add Row on Change*
4	CUR_IDX	NUMBER	*Current Record Flag*

No.	Nama Field	Tipe Data	Keterangan
5	START_DATE	DATE	*Starting Timestamp*
6	END_DATE	DATE	*Ending Timestamp*

Query yang digunakan untuk meng-*extract* data dari *data source*, meng-*transform* lalu me-*load*-nya ke *data warehouse*.

```
SELECT ffv.flex_value  AS kode_eselon,
       ffv.description AS nama_eselon
FROM   fnd_flex_value_sets ffvs,
       fnd_flex_values_vl ffv
WHERE  ffvs.flex_value_set_id=ffv.flex_value_set_id
AND    ffvs.flex_value_set_name='SPAN_DFF_ESELON_ONE';
```

Gambar 3.60 – *Query* untuk DIM_ESELON

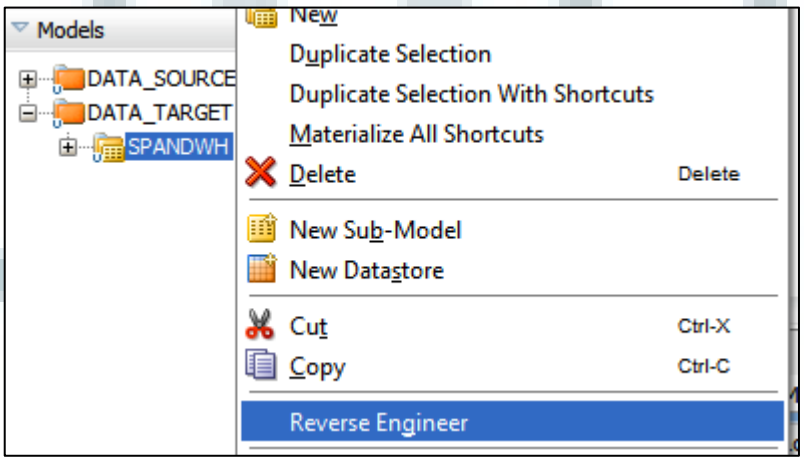
Setelah tabel target dan *query* disiapkan, dilakukan proses ETL dengan Oracle Data Integrator 11g.

a. Membuat *sequence* sebagai *Surrogate Key* pada *data warehouse.*

```
CREATE SEQUENCE DIM_ESELON_SEQ
START WITH 1 MINVALUE 1 NOMAXVALUE;
```

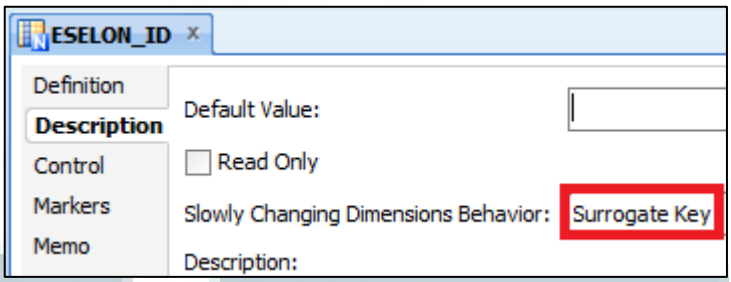
Gambar 3.61 – DDL untuk DIM_ESELON_SEQ

b. Melakukan *reverse engineer* pada Oracle Data Integrator.

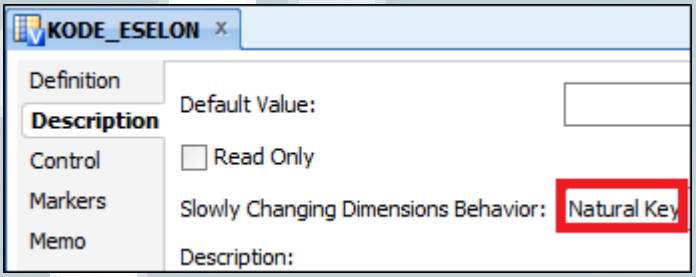


Gambar 3.62 – *Reverse Engineer* untuk DIM_ESELON

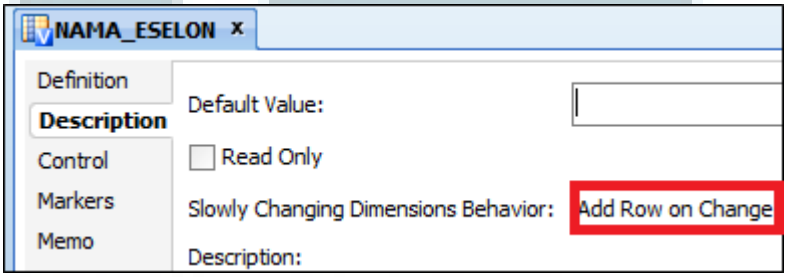
c. Mengidentifikasikan tipe *Slowly Changing Dimensions* (SCD) *Behavior* untuk setiap kolom pada tabel target.



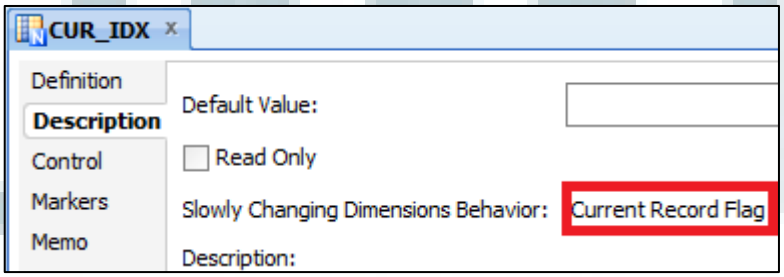
Gambar 3.63 – Spesifikasi *behavior* kolom ESELON_ID



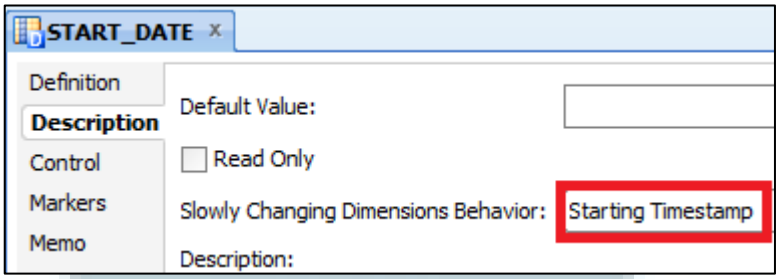
Gambar 3.64 – Spesifikasi *behavior* kolom KODE_ESELON



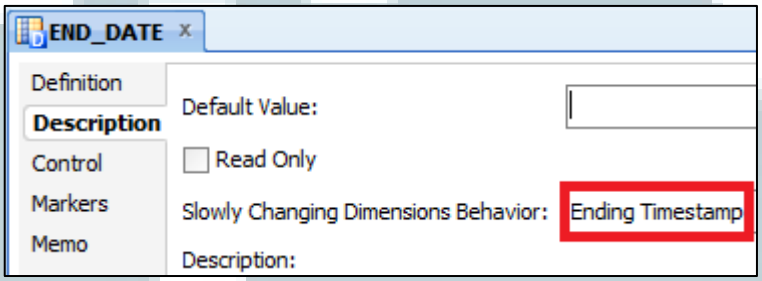
Gambar 3.65 – Spesifikasi *behavior* kolom NAMA_ESELON



Gambar 3.66 – Spesifikasi *behavior* kolom CUR_IDX

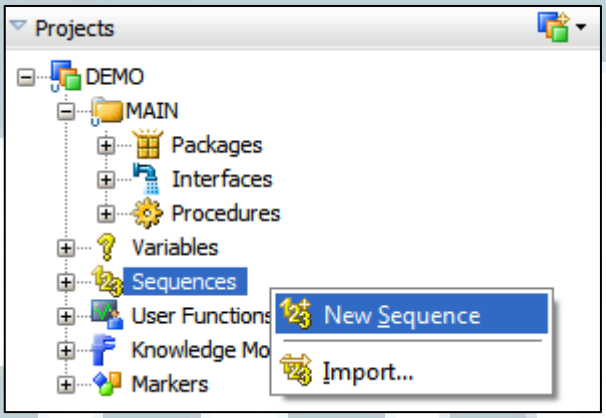


Gambar 3.67 – Spesifikasi *behavior* kolom START_DATE

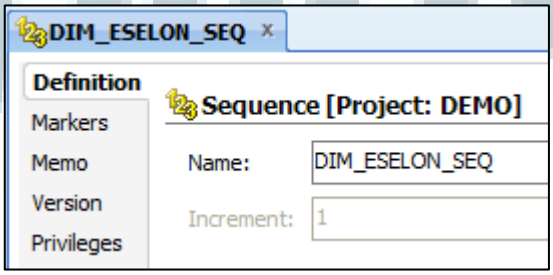


Gambar 3.68 – Spesifikasi *behavior* kolom END_DATE

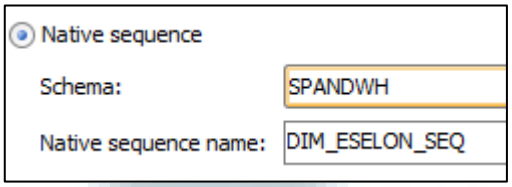
d. Membuat *project sequence* pada Oracle Data Integrator.



Gambar 3.69 – Membuat DIM_ESELON_SEQ pada ODI (Langkah 1)

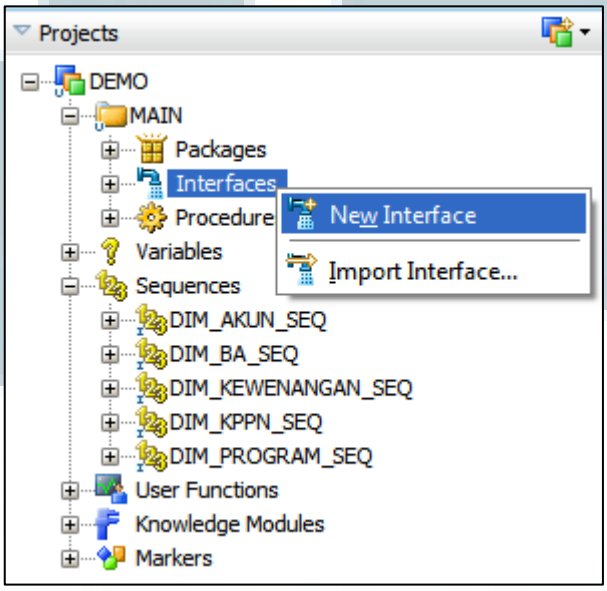


Gambar 3.70 – Membuat DIM_ESELON_SEQ pada ODI (Langkah 2)

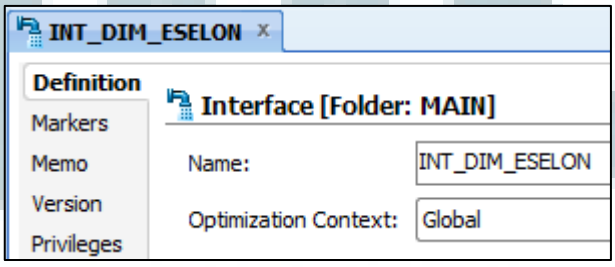


Gambar 3.71 – Membuat DIM_ESELON_SEQ pada ODI (Langkah 3)

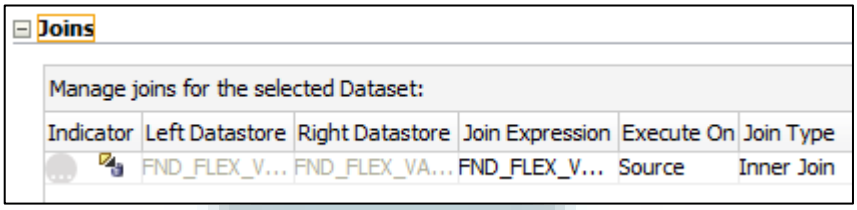
e. Melakukan *mapping* antara *data source* dan *data target* yaitu dengan membuat *interface* pada Oracle Data Integrator dimana dilakukan pengidentifikasian hubungan antara *data source* dengan *data target* yang ingin diambil.



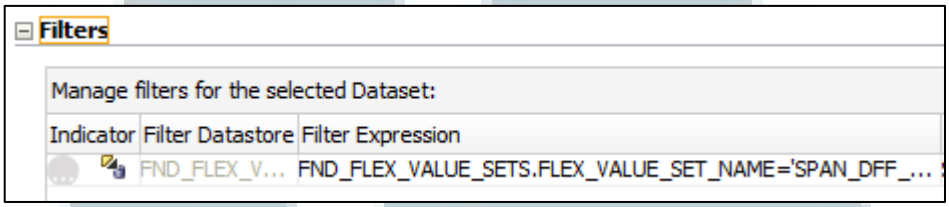
Gambar 3.72 – Membuat INT_DIM_ESELON (Langkah 1)



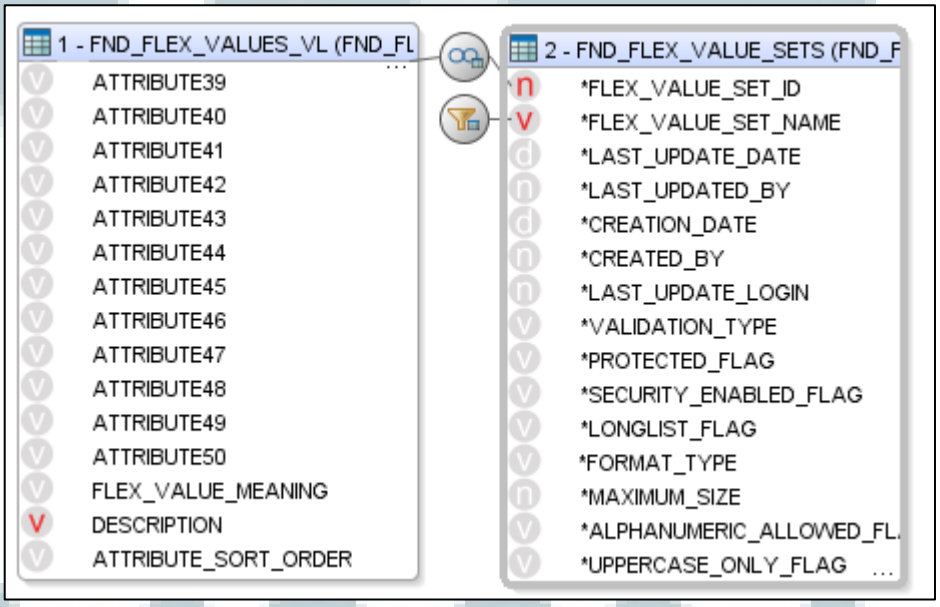
Gambar 3.73 – Membuat INT_DIM_ESELON (Langkah 2)



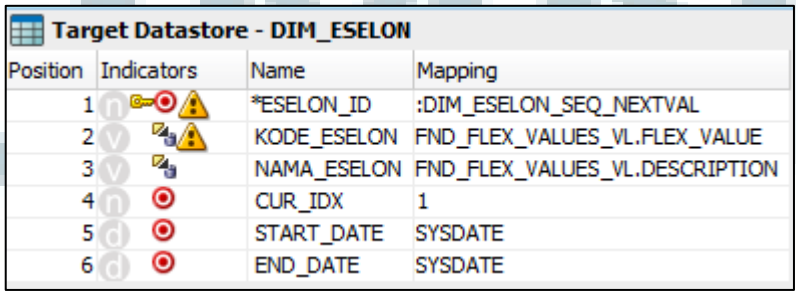
Gambar 3.74 – Membuat INT_DIM_ESELON (Langkah 3)



Gambar 3.75 – Membuat INT_DIM_ESELON (Langkah 4)

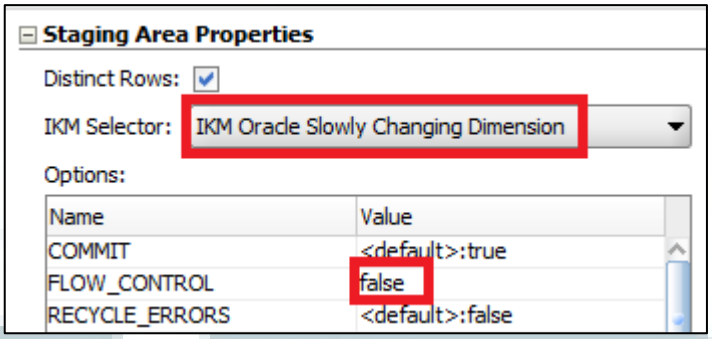


Gambar 3.76 – Membuat INT_DIM_ESELON (Langkah 5)



Gambar 3.77 – *Mapping* antara *data source* dan DIM_ESELON

f. Melakukan pengaturan *staging area*.



Gambar 3.78 – Mengatur *knowledge base* dan *flow control*

g. Menjalankan (*run*) *interface* yang telah dibuat, sehingga proses ETL akan berjalan dan data akan masuk dari *data source* ke *data target*.

Gambar 3.79 – Menjalankan INT_DIM_ESELON

5. Proses ETL pada DIM_KPPN

Tabel 3.11 – Struktur Tabel DIM_KPPN

No.	Nama Field	Tipe Data	Keterangan
1	KPPN_ID	NUMBER	*Surrogate Key (PK)*
2	KODE_KPPN	VARCHAR2 (10 Byte)	*Natural Key*
3	NAMA_KPPN	VARCHAR2 (240 Byte)	*Add Row on Change*
4	CUR_IDX	NUMBER	*Current Record Flag*

No.	Nama Field	Tipe Data	Keterangan
5	START_DATE	DATE	*Starting Timestamp*
6	END_DATE	DATE	*Ending Timestamp*

Query yang digunakan untuk meng-*extract* data dari *data source*, meng-*transform* lalu me-*load*-nya ke *data warehouse*.

```
SELECT ffv.flex_value  AS kode_kppn,
       ffv.description AS nama_kppn
FROM   fnd_flex_value_sets ffvs,
       fnd_flex_values_vl ffv
WHERE  ffvs.flex_value_set_id=ffv.flex_value_set_id
AND    ffvs.flex_value_set_name='SPAN_KPPN'
AND    ffv.summary_flag='N';
```

Gambar 3.80 – *Query* untuk DIM_KPPN

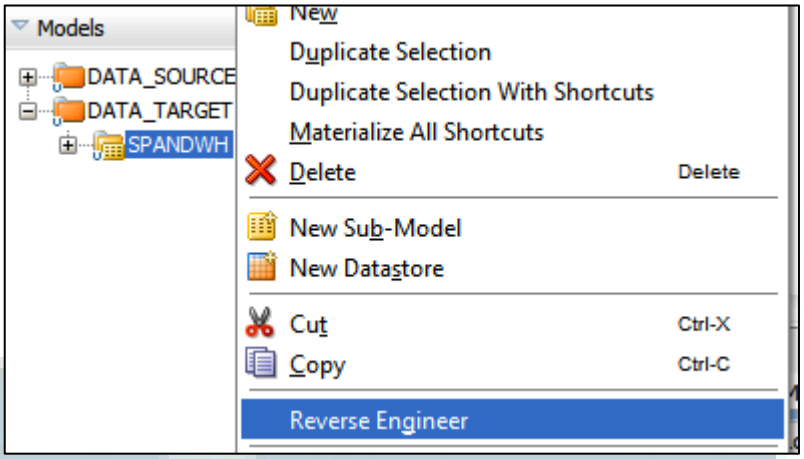
Setelah tabel target dan *query* disiapkan, dilakukan proses ETL dengan Oracle Data Integrator 11g.

a. Membuat *sequence* sebagai *Surrogate Key* pada *data warehouse.*

```
CREATE SEQUENCE DIM_KPPN_SEQ
START WITH 1 MINVALUE 1 NOMAXVALUE;
```

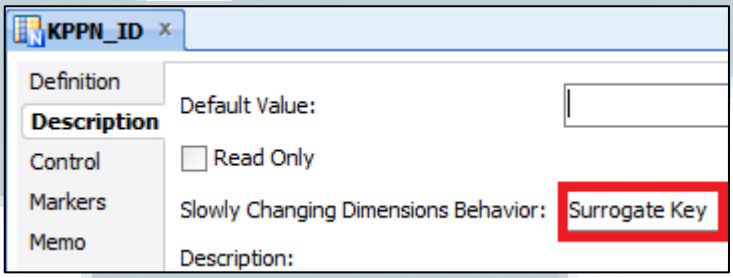
Gambar 3.81 – DDL untuk DIM_KPPN_SEQ

b. Melakukan *reverse engineer* pada Oracle Data Integrator.

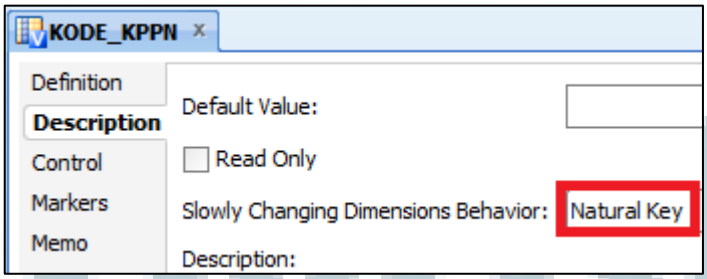


Gambar 3.82 – *Reverse Engineer* untuk DIM_KPPN

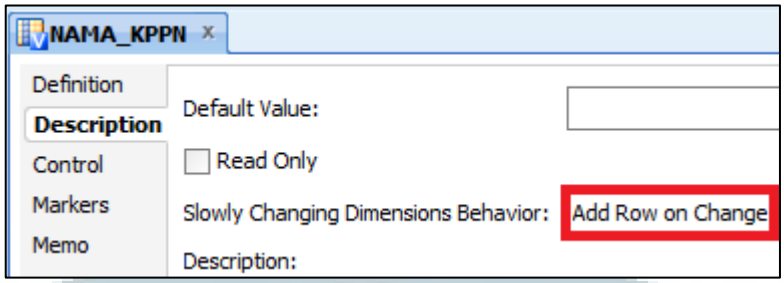
c. Mengidentifikasikan tipe *Slowly Changing Dimensions* (SCD) *Behavior* untuk setiap kolom pada tabel target.



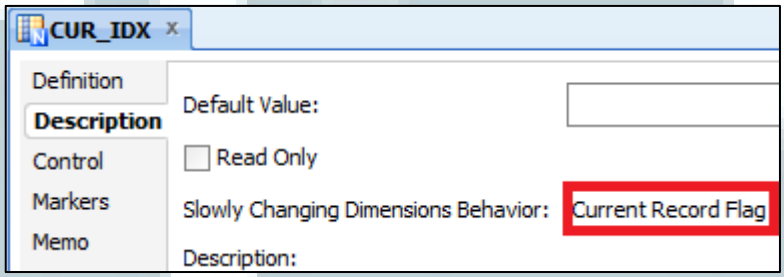
Gambar 3.83 – Spesifikasi *behavior* kolom KPPN_ID



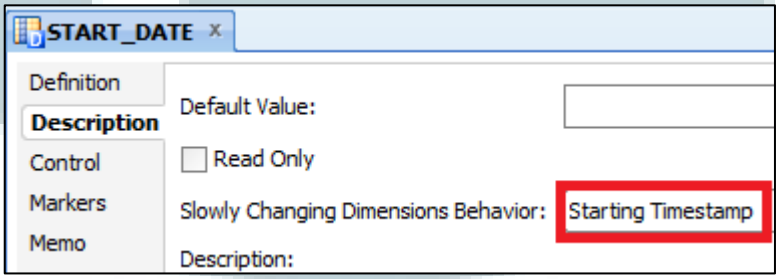
Gambar 3.84 – Spesifikasi *behavior* kolom KODE_KPPN



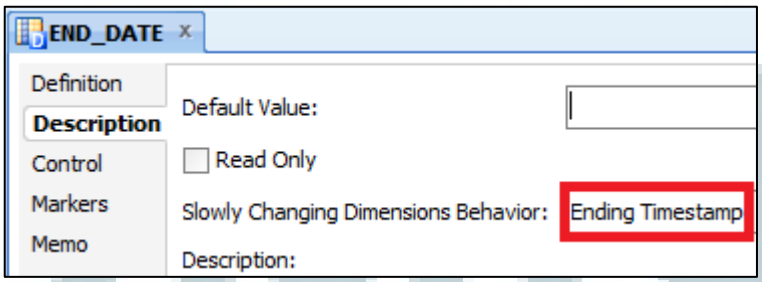
Gambar 3.85 – Spesifikasi *behavior* kolom NAMA_KPPN



Gambar 3.86 – Spesifikasi *behavior* kolom CUR_IDX

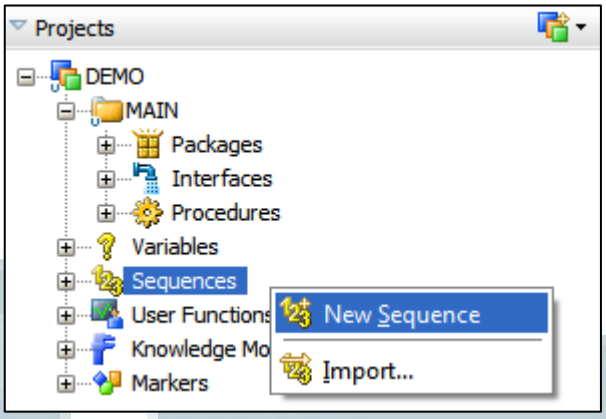


Gambar 3.87 – Spesifikasi *behavior* kolom START_DATE

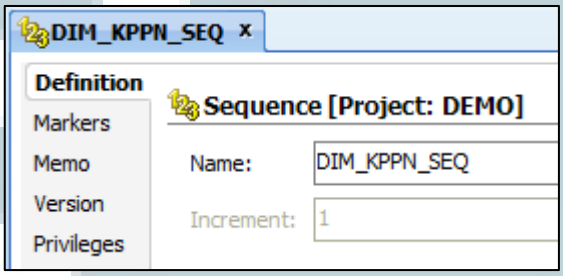


Gambar 3.88 – Spesifikasi *behavior* kolom END_DATE

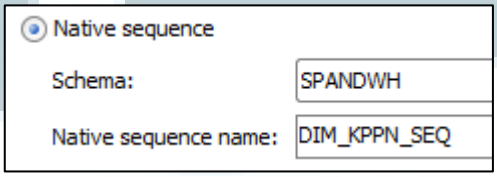
d. Membuat *project sequence* pada Oracle Data Integrator.



Gambar 3.89 – Membuat DIM_KPPN_SEQ pada ODI (Langkah 1)

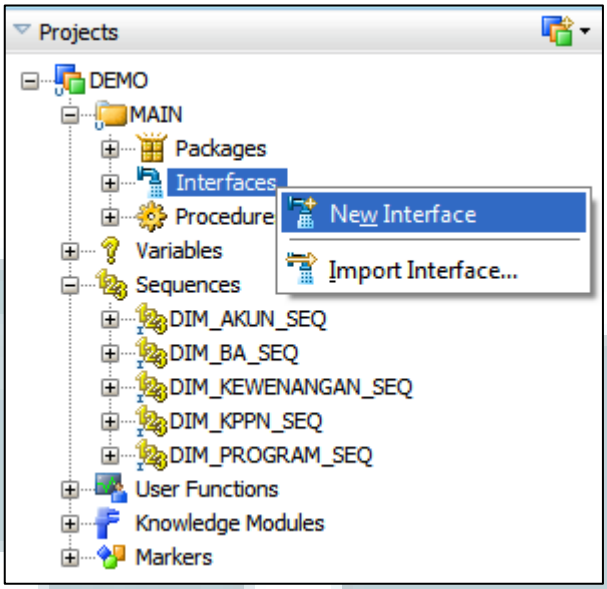


Gambar 3.90 – Membuat DIM_KPPN_SEQ pada ODI (Langkah 2)

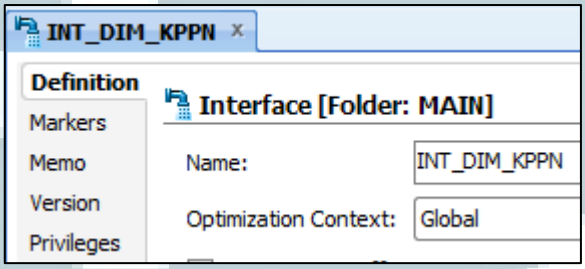


Gambar 3.91 – Membuat DIM_KPPN_SEQ pada ODI (Langkah 3)

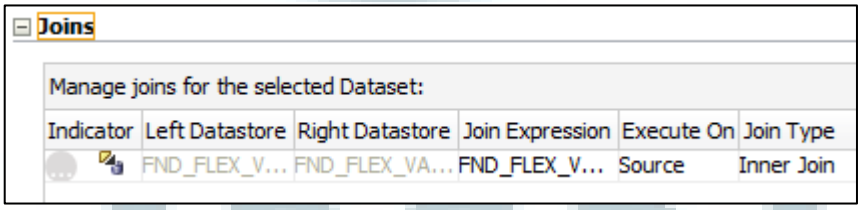
e. Melakukan *mapping* antara *data source* dan *data target* yaitu dengan membuat *interface* pada Oracle Data Integrator dimana dilakukan pengidentifikasian hubungan antara *data source* dengan *data target* yang ingin diambil.



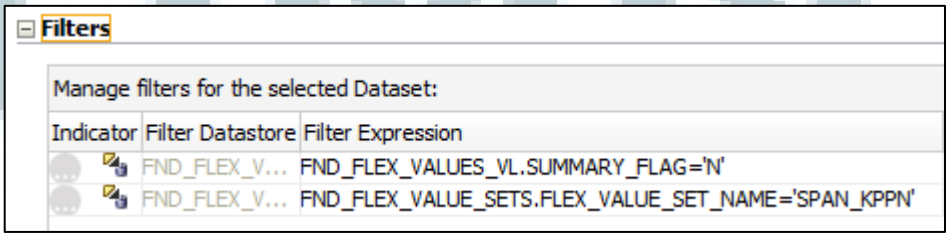
Gambar 3.92 – Membuat INT_DIM_KPPN (Langkah 1)



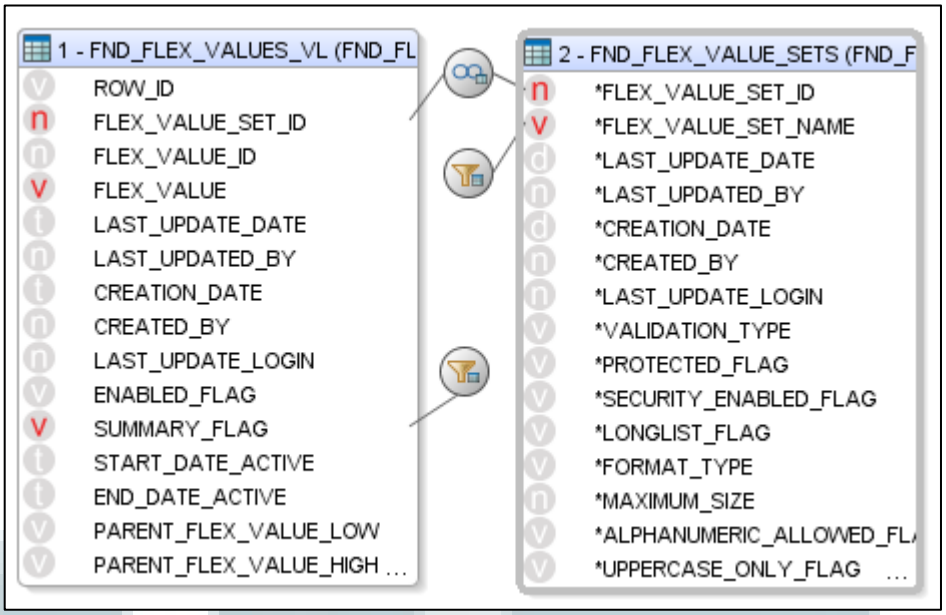
Gambar 3.93 – Membuat INT_DIM_KPPN (Langkah 2)



Gambar 3.94 – Membuat INT_DIM_KPPN (Langkah 3)



Gambar 3.95 – Membuat INT_DIM_KPPN (Langkah 4)



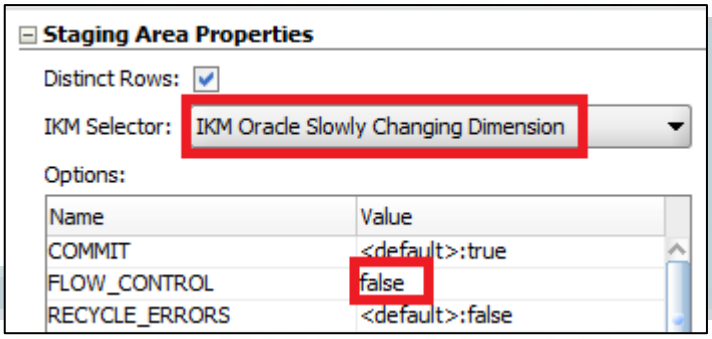
Gambar 3.96 – Membuat INT_DIM_KPPN (Langkah 5)

Target Datastore - DIM_KPPN

Position	Indicators	Name	Mapping
1		*KPPN_ID	:DIM_KPPN_SEQ_NEXTVAL
2		KODE_KPPN	FND_FLEX_VALUES_VL.FLEX_VALUE
3		NAMA_KPPN	FND_FLEX_VALUES_VL.DESCRIPTION
4		CUR_IDX	1
5		START_DATE	SYSDATE
6		END_DATE	SYSDATE

Gambar 3.97 – *Mapping* antara *data source* dan DIM_KPPN

f. Melakukan pengaturan *staging area*.



Gambar 3.98 – Mengatur *knowledge module* dan *flow control*

g. Menjalankan (*run*) *interface* yang telah dibuat, sehingga proses ETL akan berjalan dan data akan masuk dari *data source* ke *data target*.

Gambar 3.99 – Menjalankan INT_DIM_KPPN

6. Proses ETL pada DIM_SATKER

Tabel 3.12 – Struktur Tabel DIM_SATKER

No.	Nama Field	Tipe Data	Keterangan
1	SATKER_ID	NUMBER	*Surrogate Key (PK)*
2	KODE_SATKER	VARCHAR2 (10 Byte)	*Natural Key*
3	NAMA_SATKER	VARCHAR2 (240 Byte)	*Add Row on Change*
4	CUR_IDX	NUMBER	*Current Record Flag*
5	START_DATE	DATE	*Starting Timestamp*
6	END_DATE	DATE	*Ending Timestamp*

Query yang digunakan untuk meng-*extract* data dari *data source*, meng-*transform* lalu me-*load*-nya ke *data warehouse*.

```
SELECT ffv.flex_value  AS kode_satker,
       ffv.description AS nama_satker
FROM   fnd_flex_value_sets ffvs,
       fnd_flex_values_vl ffv
WHERE  ffvs.flex_value_set_id=ffv.flex_value_set_id
AND    ffvs.flex_value_set_name='SPAN_SATKER';
```

Gambar 3.100 – *Query* untuk DIM_SATKER

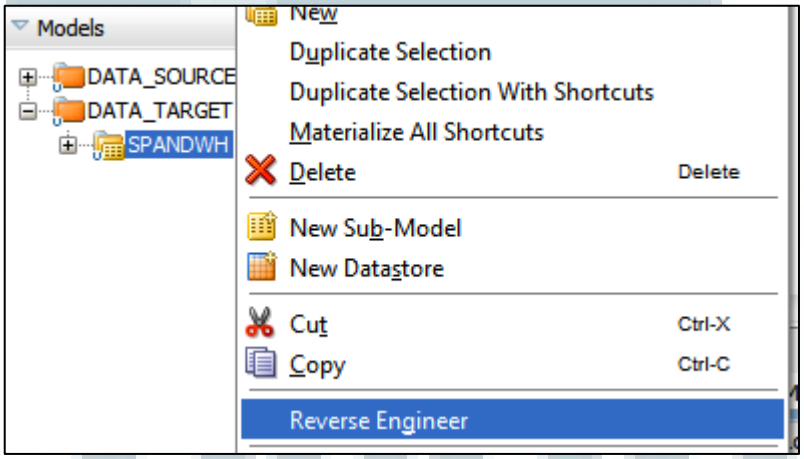
Setelah tabel target dan *query* disiapkan, dilakukan proses ETL dengan Oracle Data Integrator 11g.

a. Membuat *sequence* sebagai *Surrogate Key* pada *data warehouse.*

```
CREATE SEQUENCE DIM_SATKER_SEQ
START WITH 1 MINVALUE 1 NOMAXVALUE;
```

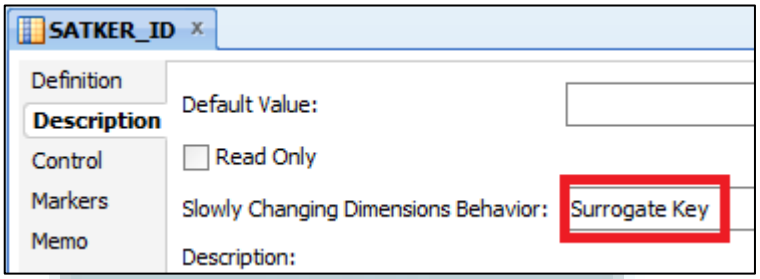
Gambar 3.101 – DDL untuk DIM_SATKER_SEQ

b. Melakukan *reverse engineer* pada Oracle Data Integrator.

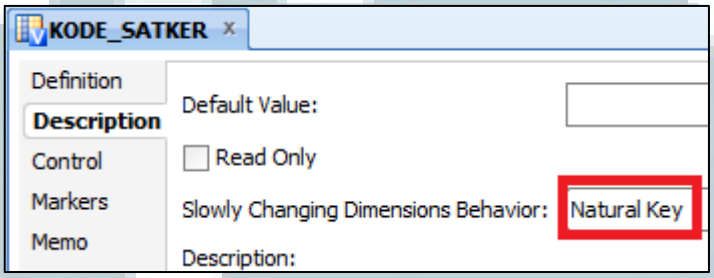


Gambar 3.102 – *Reverse Engineer* untuk DIM_SATKER

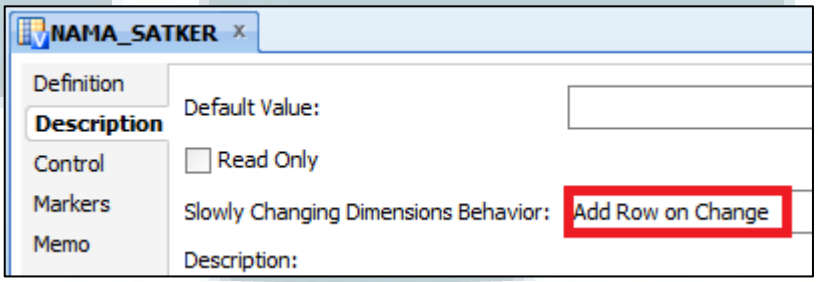
c. Mengidentifikasikan tipe *Slowly Changing Dimensions* (SCD) *Behavior* untuk setiap kolom pada tabel target.



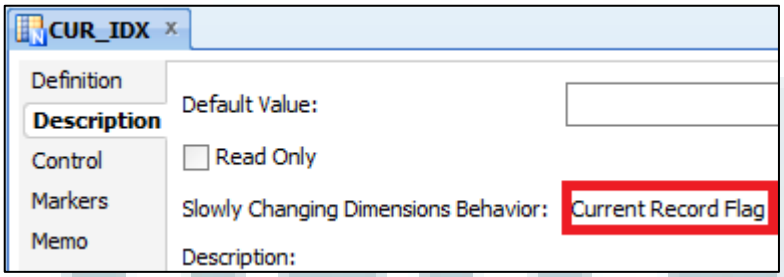
Gambar 3.103 – Spesifikasi *behavior* kolom SATKER_ID



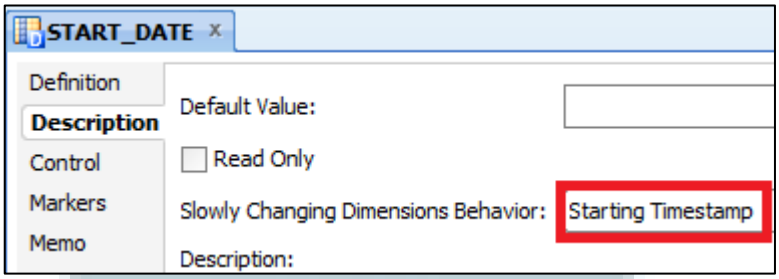
Gambar 3.104 – Spesifikasi *behavior* kolom KODE_SATKER



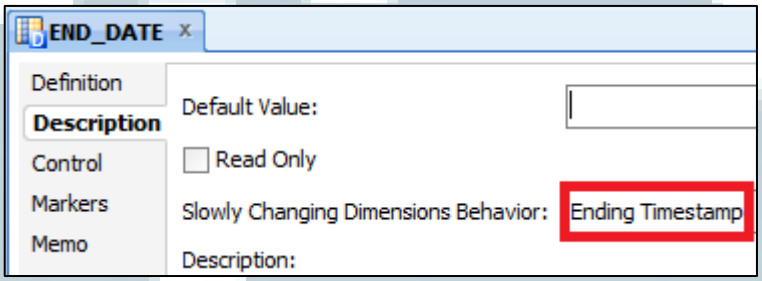
Gambar 3.105 – Spesifikasi *behavior* kolom NAMA_SATKER



Gambar 3.106 – Spesifikasi *behavior* kolom CUR_IDX

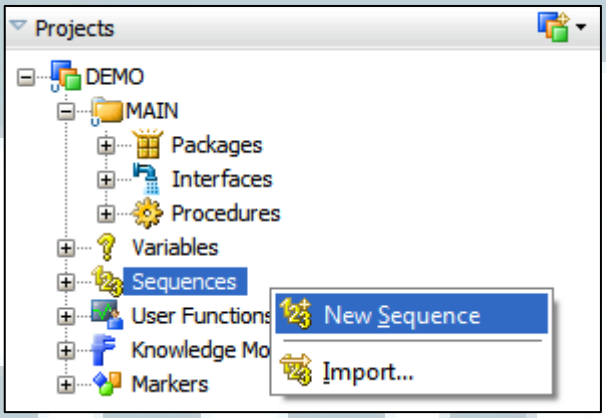


Gambar 3.107 – Spesifikasi *behavior* kolom START_DATE

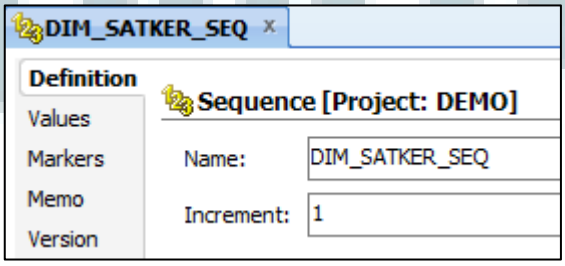


Gambar 3.108 – Spesifikasi *behavior* kolom END_DATE

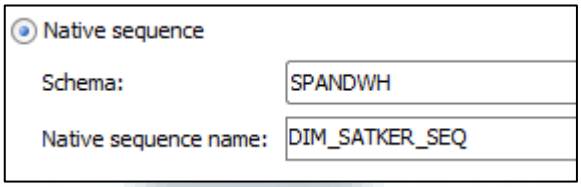
d. Membuat *project sequence* pada Oracle Data Integrator.



Gambar 3.109 – Membuat DIM_SATKER_SEQ pada ODI (Langkah 1)

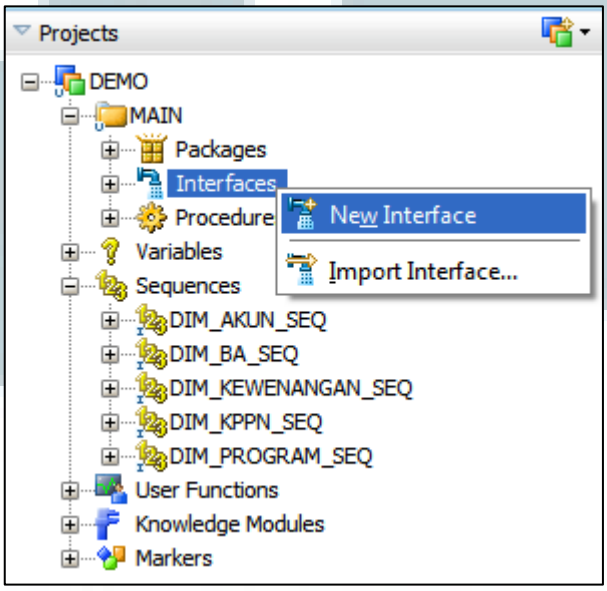


Gambar 3.110 – Membuat DIM_SATKER_SEQ pada ODI (Langkah 2)

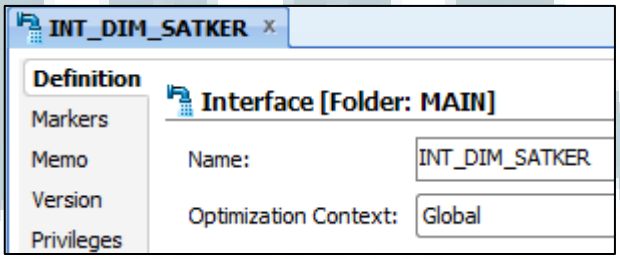


Gambar 3.111 – Membuat DIM_SATKER_SEQ pada ODI (Langkah 3)

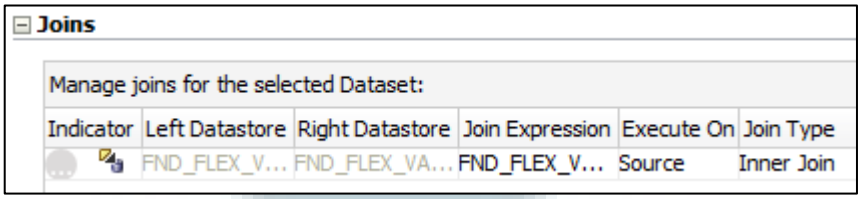
e. Melakukan *mapping* antara *data source* dan *data target* yaitu dengan membuat *interface* pada Oracle Data Integrator dimana dilakukan pengidentifikasian hubungan antara *data source* dengan *data target* yang ingin diambil.



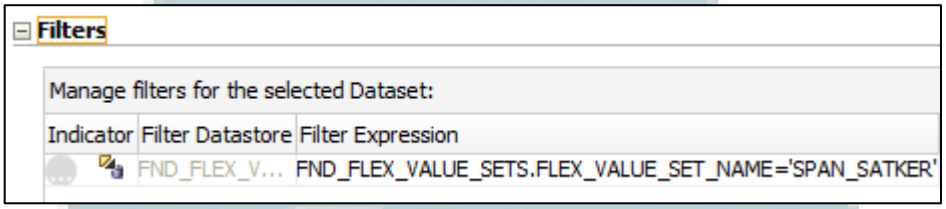
Gambar 3.112 – Membuat INT_DIM_SATKER (Langkah 1)



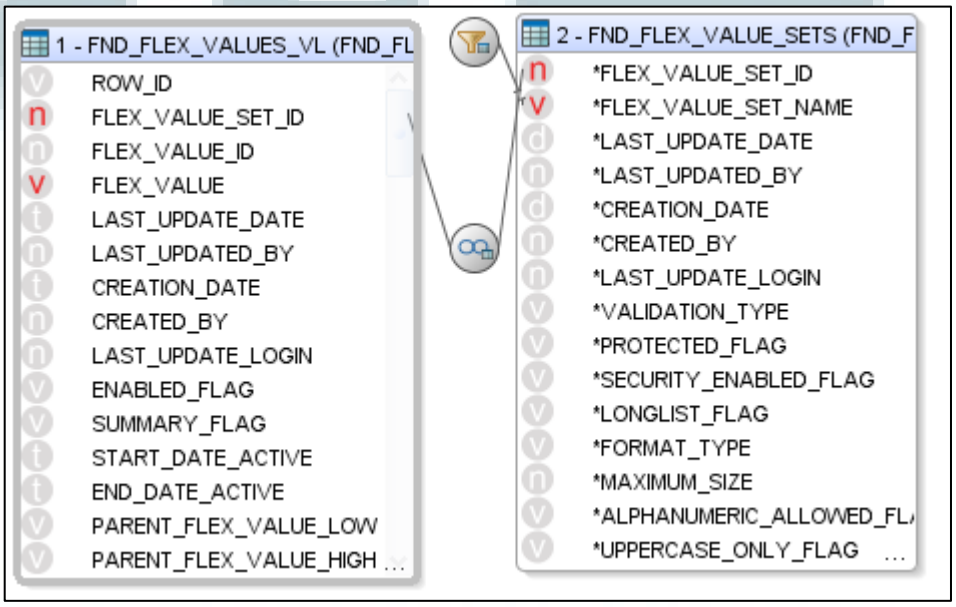
Gambar 3.113 – Membuat INT_DIM_SATKER (Langkah 2)



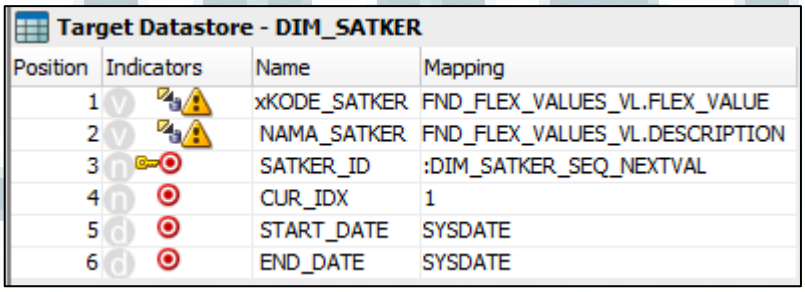
Gambar 3.114 – Membuat INT_DIM_SATKER (Langkah 3)



Gambar 3.115 – Membuat INT_DIM_SATKER (Langkah 4)

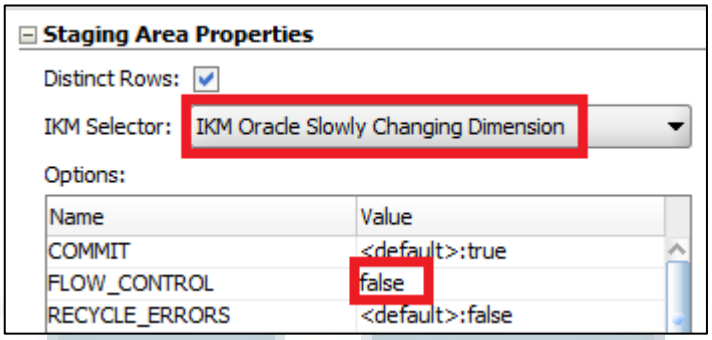


Gambar 3.116 – Membuat INT_DIM_SATKER (Langkah 5)



Gambar 3.117 – *Mapping* antara *data source* dan DIM_SATKER

f. Melakukan pengaturan *staging area*.



Gambar 3.118 – Mengatur *knowledge base* dan *flow control*

g. Menjalankan (*run*) *interface* yang telah dibuat, sehingga proses ETL akan berjalan dan data akan masuk dari *data source* ke *data target*.

Gambar 3.119 – Menjalankan INT_DIM_SATKER

7. Proses ETL pada DIM_BUDGET

Tabel 3.13 – Struktur Tabel DIM_BUDGET

No.	Nama Field	Tipe Data	Keterangan
1	BUDGET_ID	NUMBER	*Surrogate Key (PK)*
2	KODE_BUDGET	VARCHAR2 (10 Byte)	*Natural Key*
3	NAMA_BUDGET	VARCHAR2 (240 Byte)	*Add Row on Change*
4	CUR_IDX	NUMBER	*Current Record Flag*
5	START_DATE	DATE	*Starting Timestamp*

No.	Nama Field	Tipe Data	Keterangan
6	END_DATE	DATE	*Ending Timestamp*

Query yang digunakan untuk meng-*extract* data dari *data source*, meng-*transform* lalu me-*load*-nya ke *data warehouse.*

```
SELECT ffv.flex_value  AS kode_budget,
       ffv.description AS nama_budget
FROM   fnd_flex_value_sets ffvs,
       fnd_flex_values_vl ffv
WHERE  ffvs.flex_value_set_id=ffv.flex_value_set_id
AND    ffvs.flex_value_set_name='SPAN_BUDGET_TYPE'
AND    ffv.summary_flag='N';
```

Gambar 3.120 – *Query* untuk DIM_BUDGET

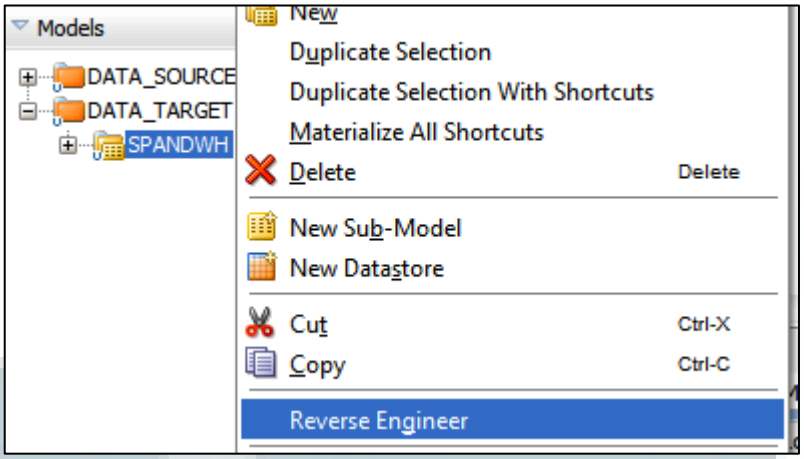
Setelah tabel target dan *query* disiapkan, dilakukan proses ETL dengan Oracle Data Integrator 11g.

a. Membuat *sequence* sebagai *Surrogate Key* pada *data warehouse.*

```
CREATE SEQUENCE DIM_BUDGET_SEQ
START WITH 1 MINVALUE 1 NOMAXVALUE;
```

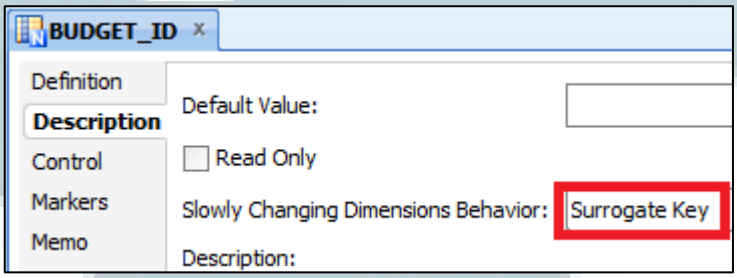
Gambar 3.121 – DDL untuk DIM_BUDGET_SEQ

b. Melakukan *reverse engineer* pada Oracle Data Integrator.

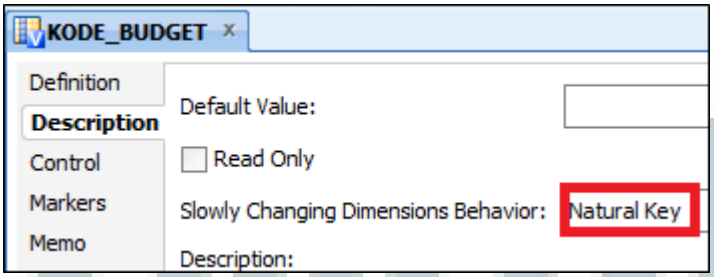


Gambar 3.122 – *Reverse Engineer* untuk DIM_BUDGET

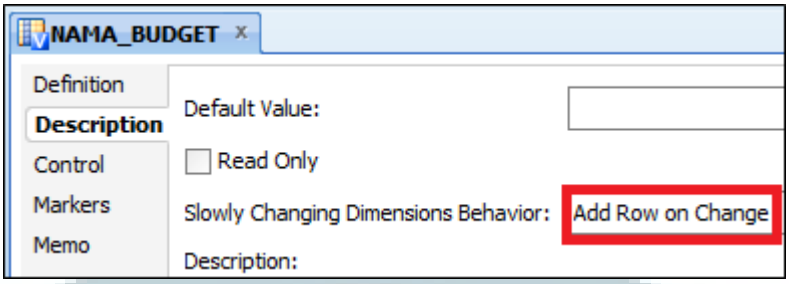
c. Mengidentifikasikan tipe *Slowly Changing Dimensions* (SCD) *Behavior* untuk setiap kolom pada tabel target.



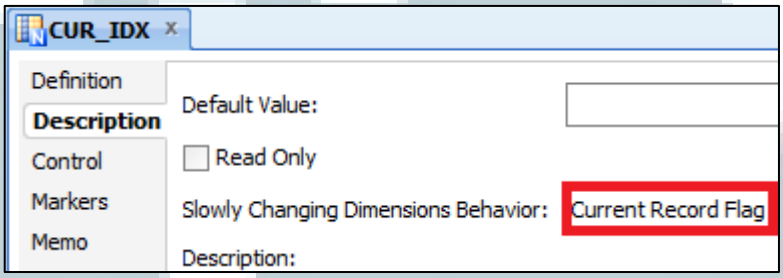
Gambar 3.123 – Spesifikasi *behavior* kolom BUDGET_ID



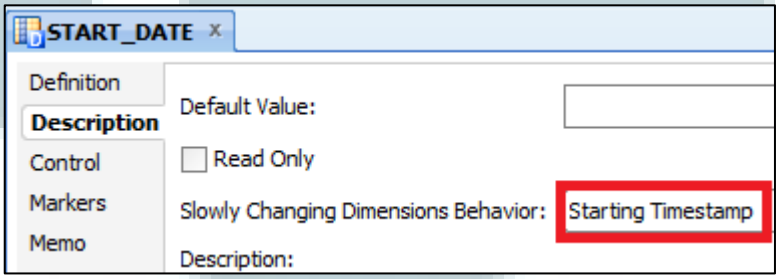
Gambar 3.124 – Spesifikasi *behavior* kolom KODE_BUDGET



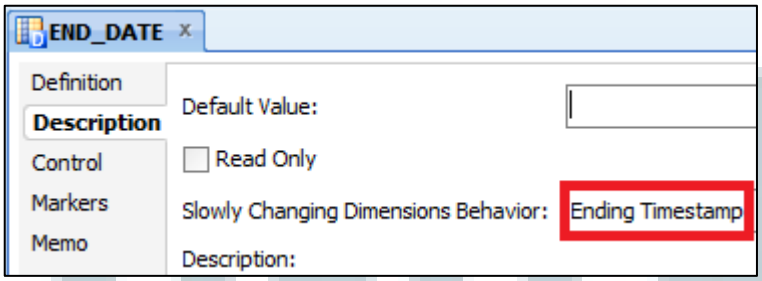
Gambar 3.125 – Spesifikasi *behavior* kolom NAMA_BUDGET



Gambar 3.126 – Spesifikasi *behavior* kolom CUR_IDX

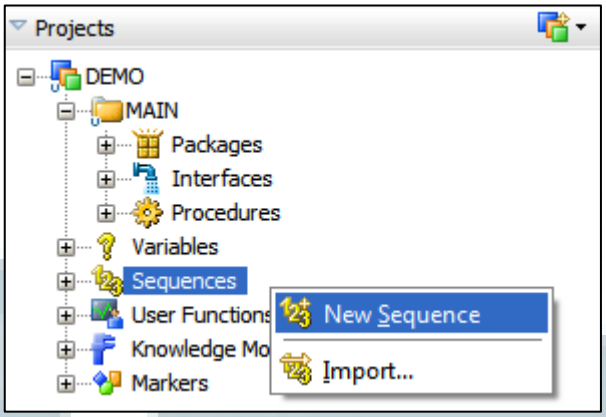


Gambar 3.127 – Spesifkasi *behavior* kolom START_DATE

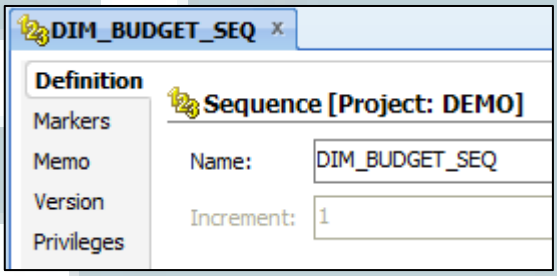


Gambar 3.128 – Spesifikasi *behavior* kolom END_DATE

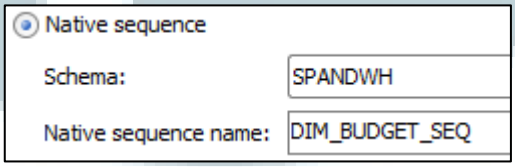
d. Membuat *project sequence* pada Oracle Data Integrator.



Gambar 3.129 – Membuat DIM_BUDGET_SEQ pada ODI (Langkah 1)

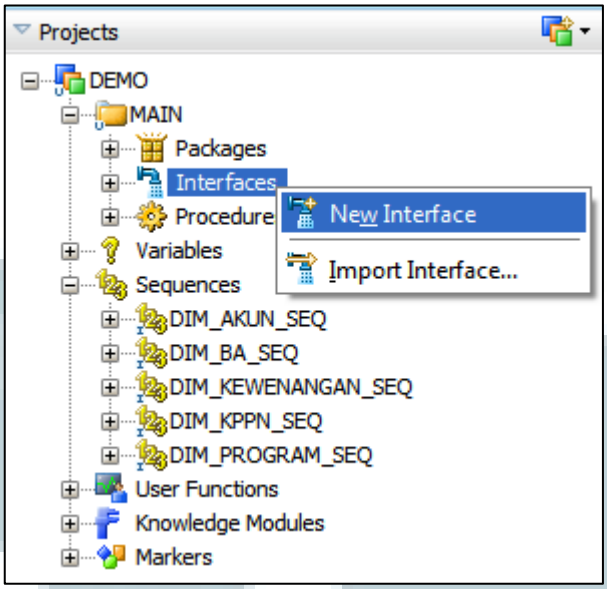


Gambar 3.130 – Membuat DIM_BUDGET_SEQ pada ODI (Langkah 2)

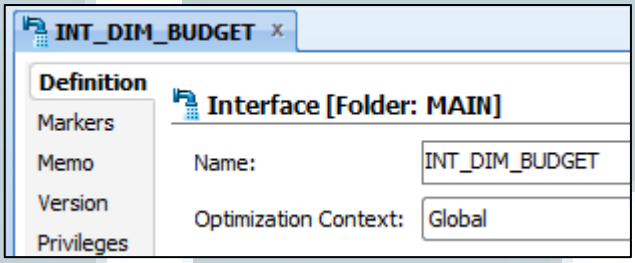


Gambar 3.131 – Membuat DIM_BUDGET_SEQ pada ODI (Langkah 3)

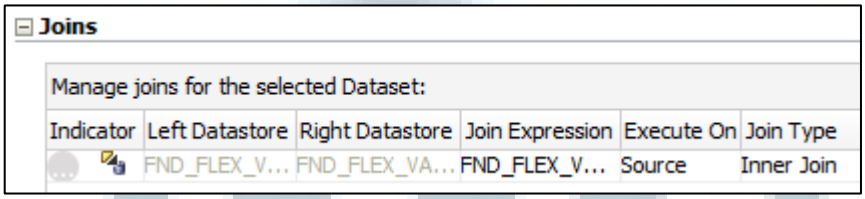
e. Melakukan *mapping* antara *data source* dan *data target* yaitu dengan membuat *interface* pada Oracle Data Integrator dimana dilakukan pengidentifikasian hubungan antara *data source* dengan *data target* yang ingin diambil.



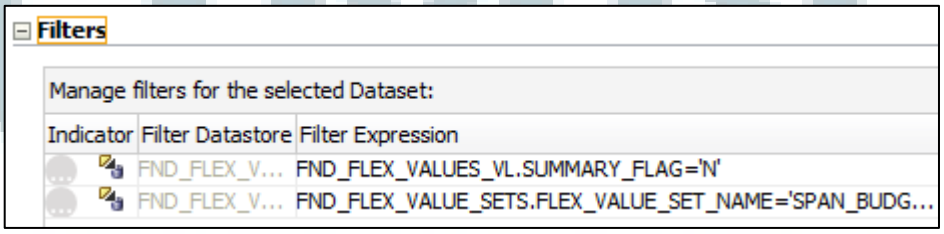
Gambar 3.132 – Membuat INT_DIM_BUDGET (Langkah 1)



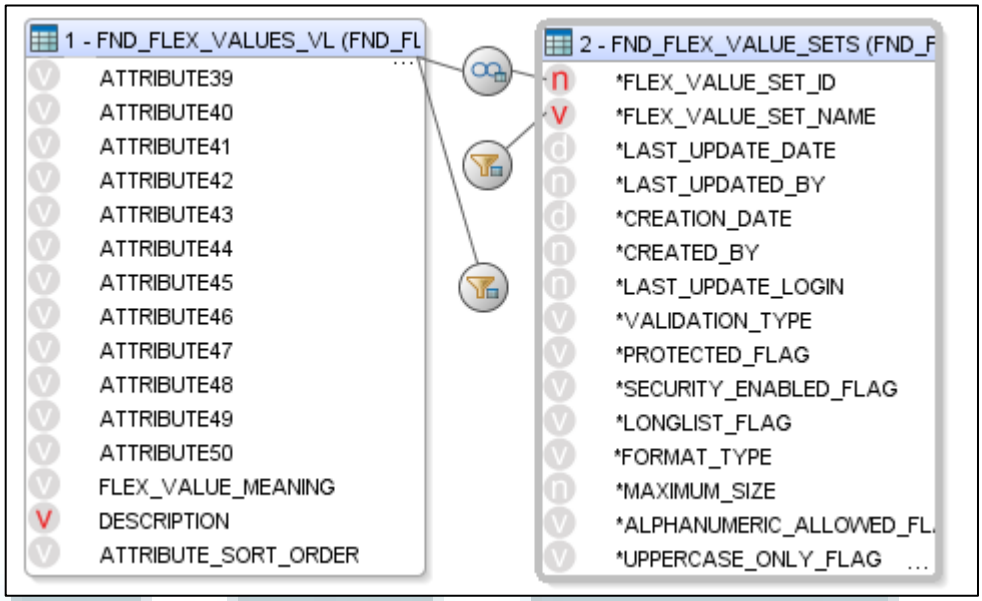
Gambar 3.133 – Membuat INT_DIM_BUDGET (Langkah 2)



Gambar 3.134 – Membuat INT_DIM_BUDGET (Langkah 3)



Gambar 3.135 – Membuat INT_DIM_BUDGET (Langkah 4)



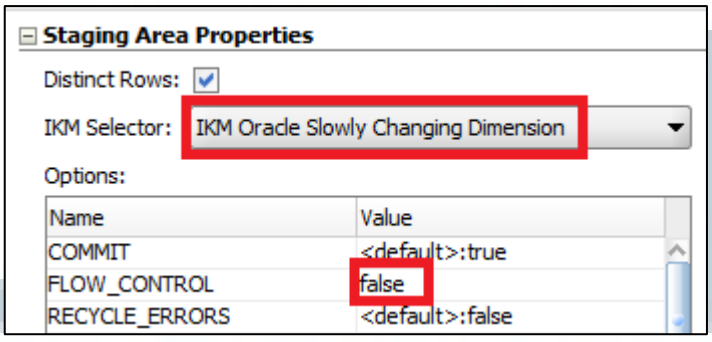
Gambar 3.136 – Membuat INT_DIM_BUDGET (Langkah 5)

Target Datastore - DIM_BUDGET

Position	Indicators	Name	Mapping
1		*BUDGET_ID	:DIM_BUDGET_SEQ_NEXTVAL
2		KODE_BUDGET	FND_FLEX_VALUES_VL.FLEX_VALUE
3		NAMA_BUDGET	FND_FLEX_VALUES_VL.DESCRIPTION
4		CUR_IDX	1
5		START_DATE	SYSDATE
6		END_DATE	SYSDATE

Gambar 3.137 – *Mapping* antara *data source* dan DIM_BUDGET

f. Melakukan pengaturan *staging area*.



Gambar 3.138 – Mengatur *knowledge base* dan *flow control*

g. Menjalankan (*run*) *interface* yang telah dibuat, sehingga proses ETL akan berjalan dan data akan masuk dari *data source* ke *data target*.

Gambar 3.139 – Menjalankan INT_DIM_BUDGET

8. Proses ETL pada DIM_LOKASI

Tabel 3.14 – Struktur Tabel DIM_LOKASI

No.	Nama Field	Tipe Data	Keterangan
1	LOKASI_ID	NUMBER	*Surrogate Key (PK)*
2	KODE_LOKASI	VARCHAR2 (10 Byte)	*Natural Key*
3	NAMA_LOKASI	VARCHAR2 (240 Byte)	*Add Row on Change*
4	CUR_IDX	NUMBER	*Current Record Flag*
5	START_DATE	DATE	*Starting Timestamp*
6	END_DATE	DATE	*Ending Timestamp*

Query yang digunakan untuk meng-*extract* data dari *data source*, meng-*transform* lalu me-*load*-nya ke *data warehouse*.

```
SELECT ffv.flex_value  AS kode_lokasi,
       ffv.description AS nama_lokasi
FROM   fnd_flex_value_sets ffvs,
       fnd_flex_values_vl ffv
WHERE  ffvs.flex_value_set_id=ffv.flex_value_set_id
AND    ffvs.flex_value_set_name='SPAN_LOKASI';
```

Gambar 3.140 – *Query* untuk DIM_LOKASI

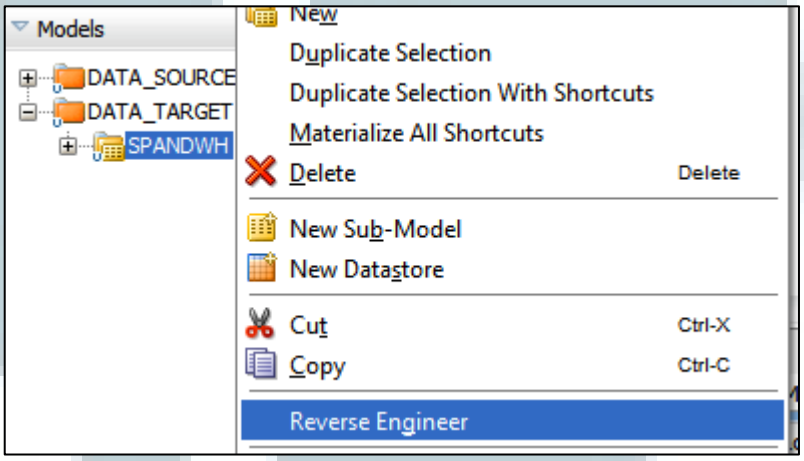
Setelah tabel target dan *query* disiapkan, dilakukan proses ETL dengan Oracle Data Integrator 11g.

a. Membuat *sequence* sebagai *Surrogate Key* pada *data warehouse.*

```
CREATE SEQUENCE DIM_LOKASI_SEQ
START WITH 1 MINVALUE 1 NOMAXVALUE;
```

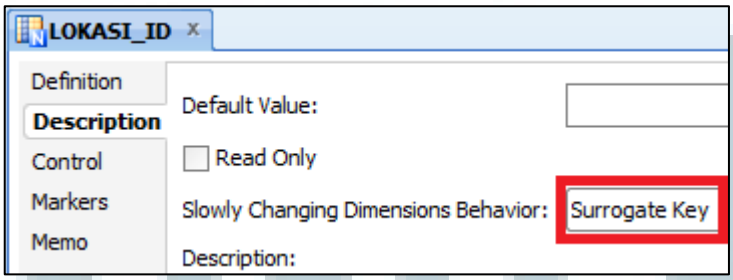
Gambar 3.141 – DDL untuk DIM_LOKASI_SEQ

b. Melakukan *reverse engineer* pada Oracle Data Integrator.

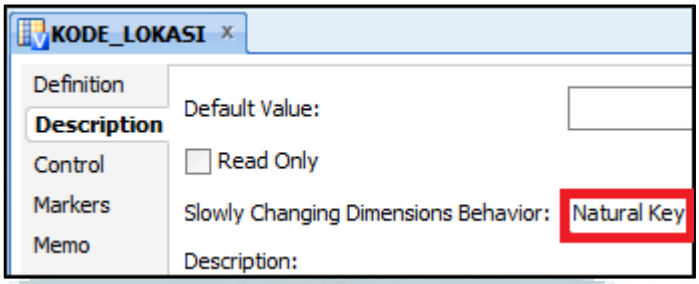


Gambar 3.142 – *Reverse Engineer* untuk DIM_LOKASI

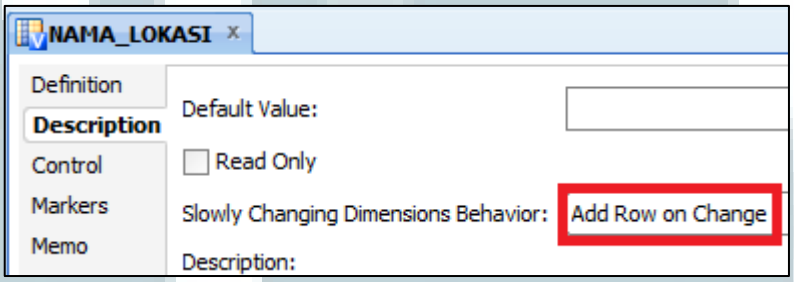
c. Mengidentifikasikan tipe *Slowly Changing Dimensions* (SCD) *Behavior* untuk setiap kolom pada tabel target.



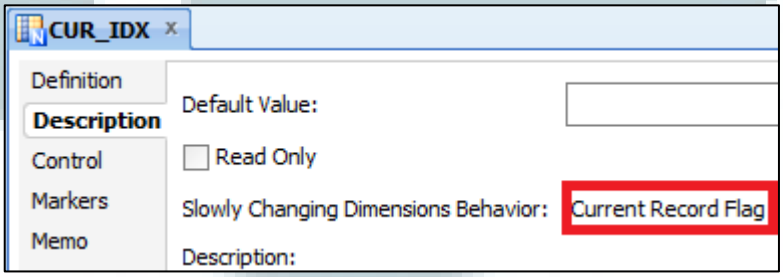
Gambar 3.143 – Spesifikasi *behavior* kolom LOKASI_ID



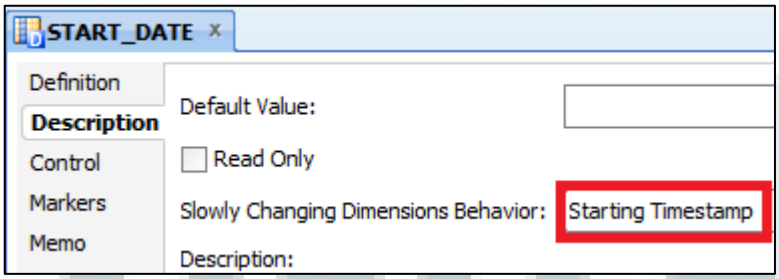
Gambar 3.144 – Spesifikasi *behavior* kolom KODE_LOKASI



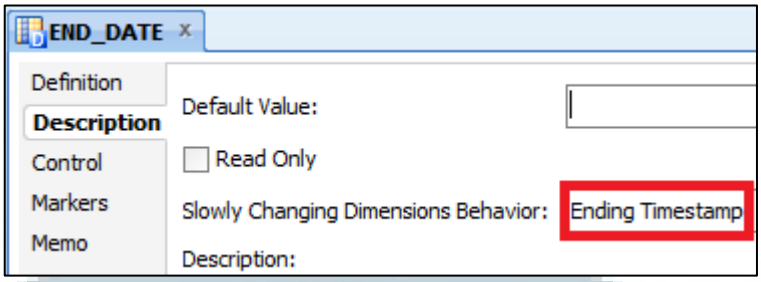
Gambar 3.145 – Spesifikasi *behavior* kolom NAMA_LOKASI



Gambar 3.146 – Spesifikasi *behavior* kolom CUR_IDX

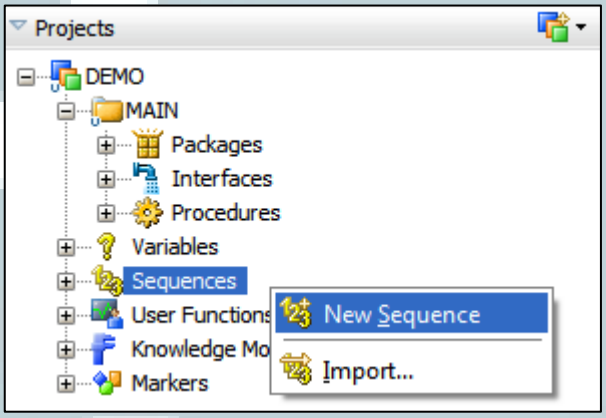


Gambar 3.147 – Spesifikasi *behavior* kolom START_DATE

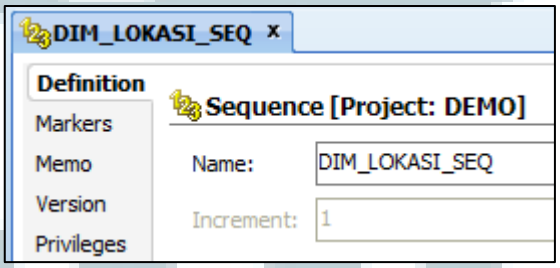


Gambar 3.148 – Spesifikasi *behavior* kolom END_DATE

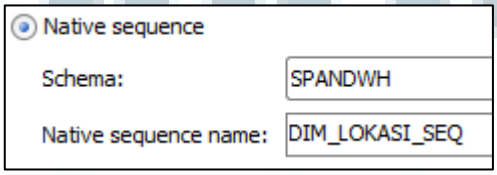
d. Membuat *project sequence* pada Oracle Data Integrator.



Gambar 3.149 – Membuat DIM_LOKASI_SEQ pada ODI (Langkah 1)



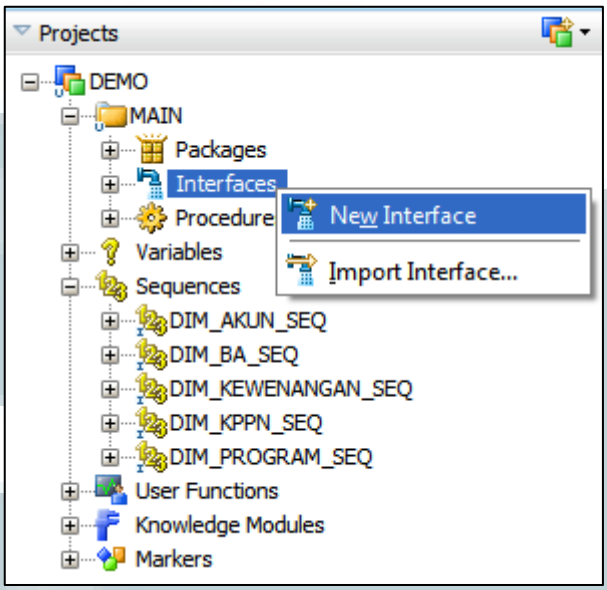
Gambar 3.150 – Membuat DIM_LOKASI_SEQ pada ODI (Langkah 2)



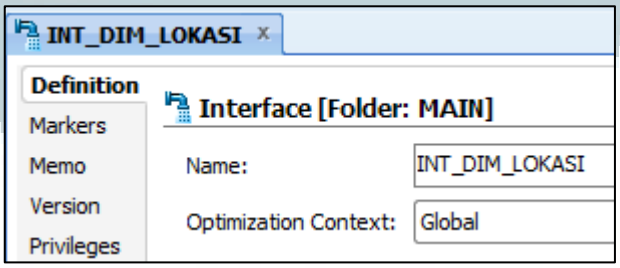
Gambar 3.151 – Membuat DIM_LOKASI_SEQ pada ODI (Langkah 3)

e. Melakukan *mapping* antara *data source* dan *data target* yaitu dengan membuat *interface* pada Oracle Data Integrator dimana dilakukan

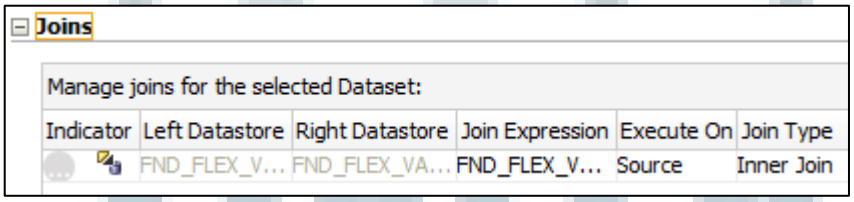
pengidentifikasian hubungan antara *data source* dengan *data target* yang ingin diambil.



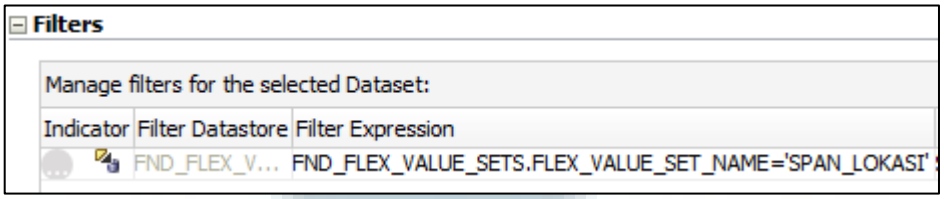
Gambar 3.152 – Membuat INT_DIM_LOKASI (Langkah 1)



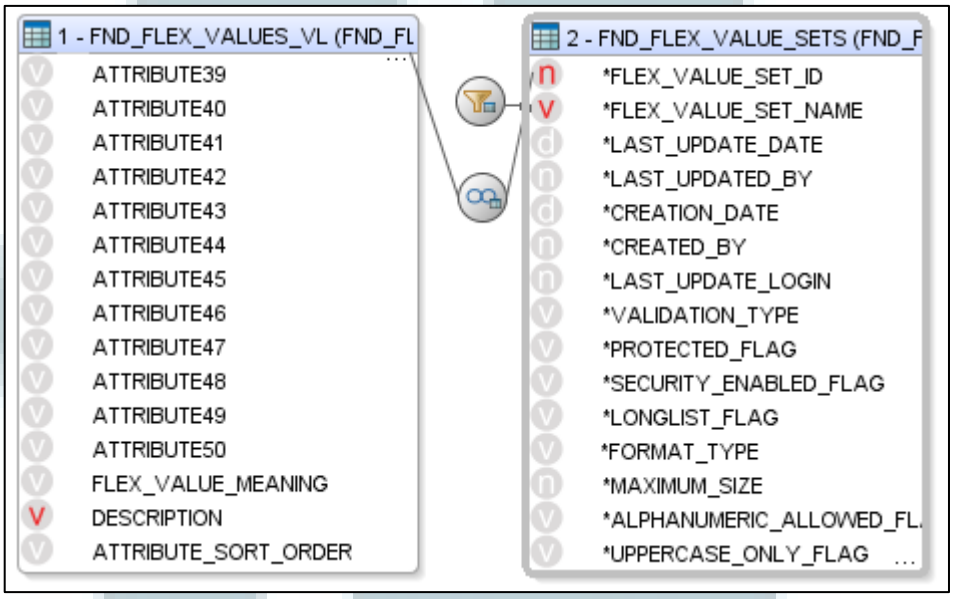
Gambar 3.153 – Membuat INT_DIM_LOKASI (Langkah 2)



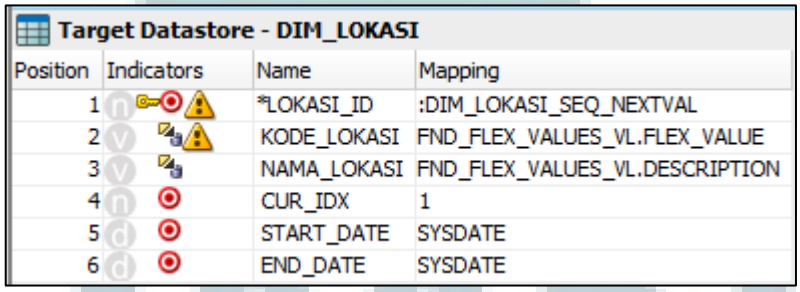
Gambar 3.154 – Membuat INT_DIM_LOKASI (Langkah 3)



Gambar 3.155 – Membuat INT_DIM_LOKASI (Langkah 4)

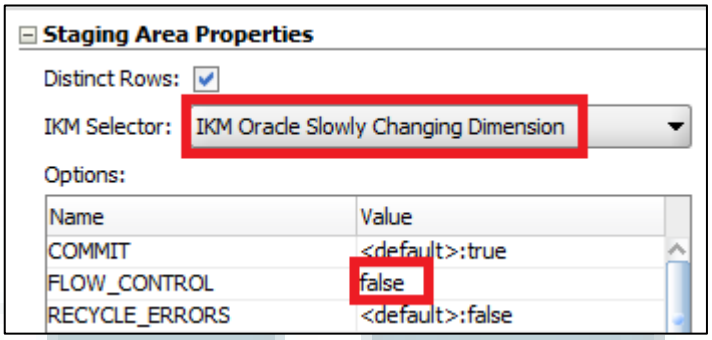


Gambar 3.156 – Membuat INT_DIM_LOKASI (Langkah 5)



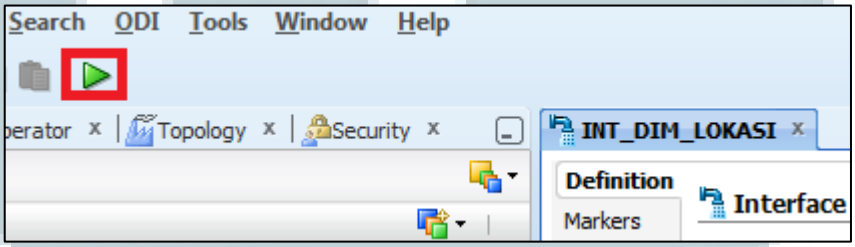
Gambar 3.157 – *Mapping* antara *data source* dan DIM_LOKASI

f. Melakukan pengaturan *staging area*.



Gambar 3.158 – Mengatur *knowledge base* dan *flow control*

g. Menjalankan (*run*) *interface* yang telah dibuat, sehingga proses ETL akan berjalan dan data akan masuk dari *data source* ke *data target*.

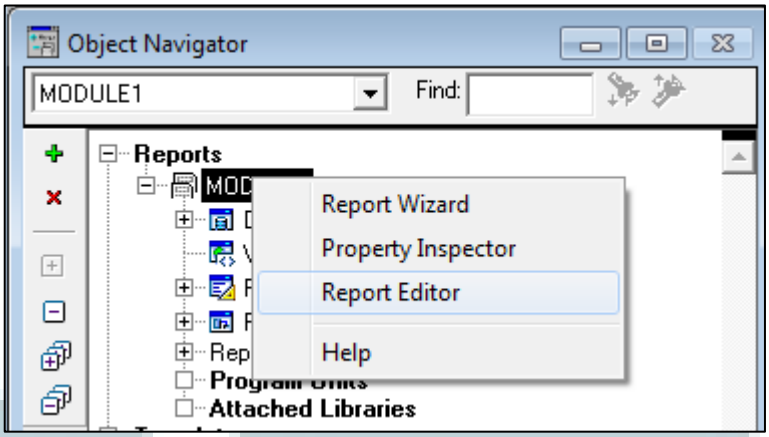


Gambar 3.159 – Menjalankan INT_DIM_LOKASI

9. Proses Pengembangan Report SPGLR00276

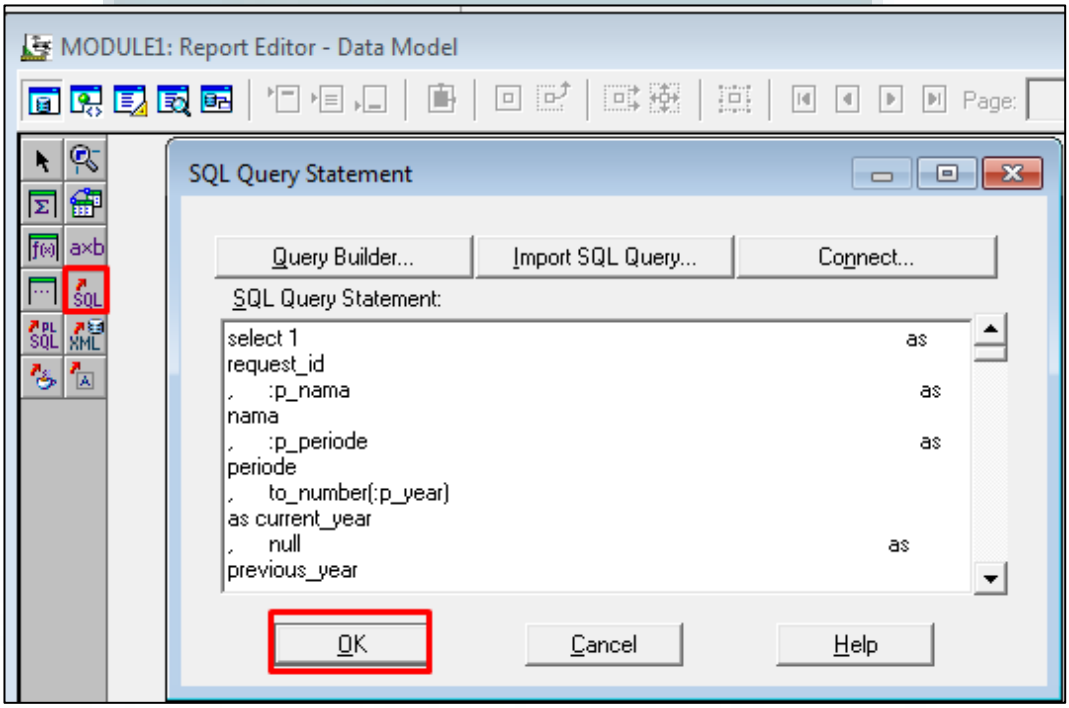
Proses pengembangan *report* menggunakan *Oracle Develper Suite* yaitu *Oracle Forms* dan *Oracle Reports* serta menggunakan *add-in Oracle BI Publisher* pada Microsoft Word. Data untuk *report* ini diambil dari *fact table* hasil ETL yang dibuat oleh anggota tim *developer* lain serta dengan *dimension table* dan *query* yang dibuat oleh peserta magang. Langkah-langkah pengembangan *report* "Neraca Tingkat KPPN" (SPGLR00276).

a. Pada *Oracle Report*, klik kanan pada modul yang menjadi tempat pengembangan *report*, lalu pilih "*Report Editor*".



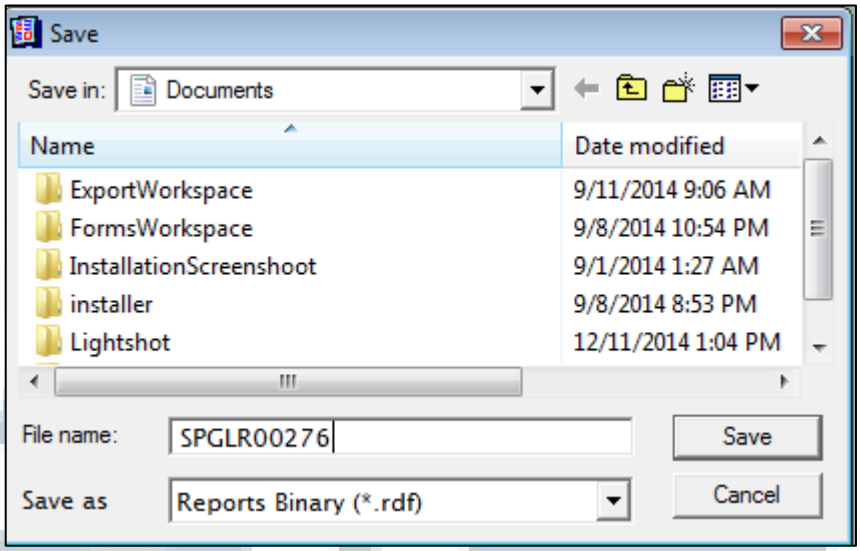
Gambar 3.160 – Membuka *Report Editor*

b. Setelah "*Report Editor*" muncul, klik tombol "SQL" lalu masukkan *query* yang telah dibuat ke tempat yang telah disediakan dan klik tombol "OK". Demi kerahasiaan data, *query* tersebut tidak dapat dilampirkan dalam laporan ini.



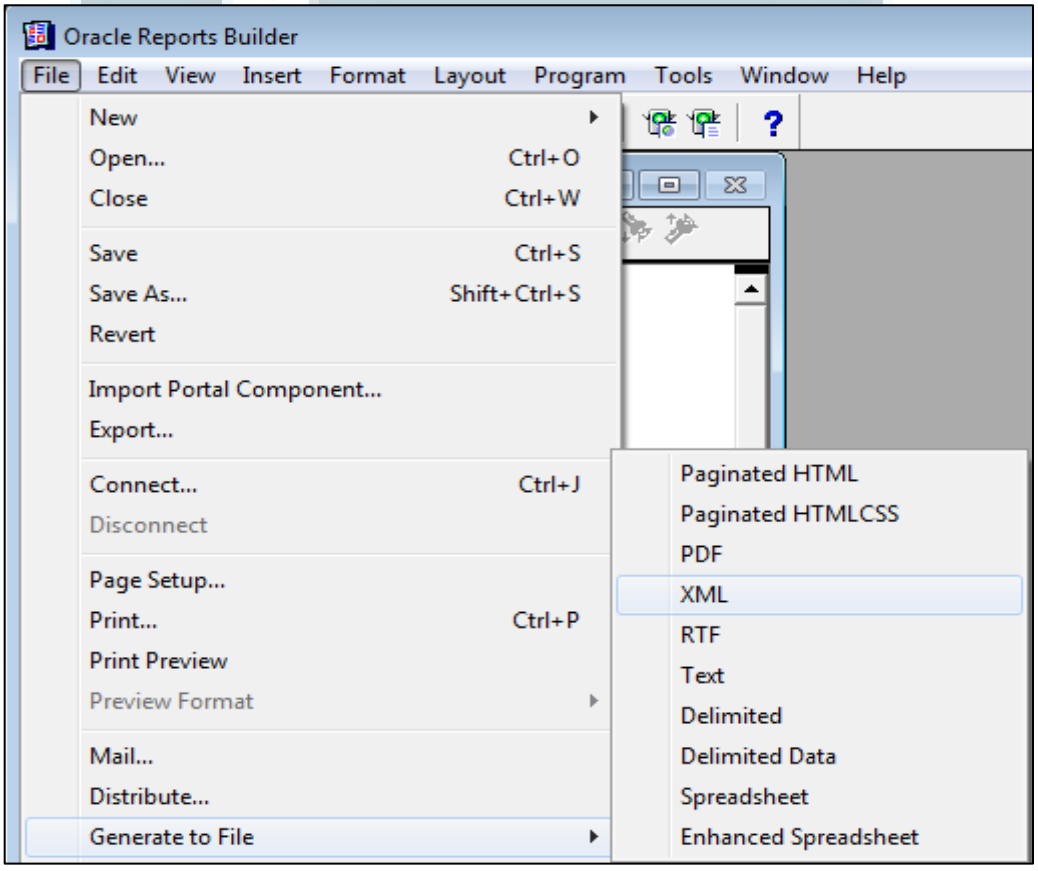
Gambar 3.161 – Memasukkan *query* ke *Report Editor*

c. Simpan modul yang telah berisi *query* dalam *file* *.rdf.

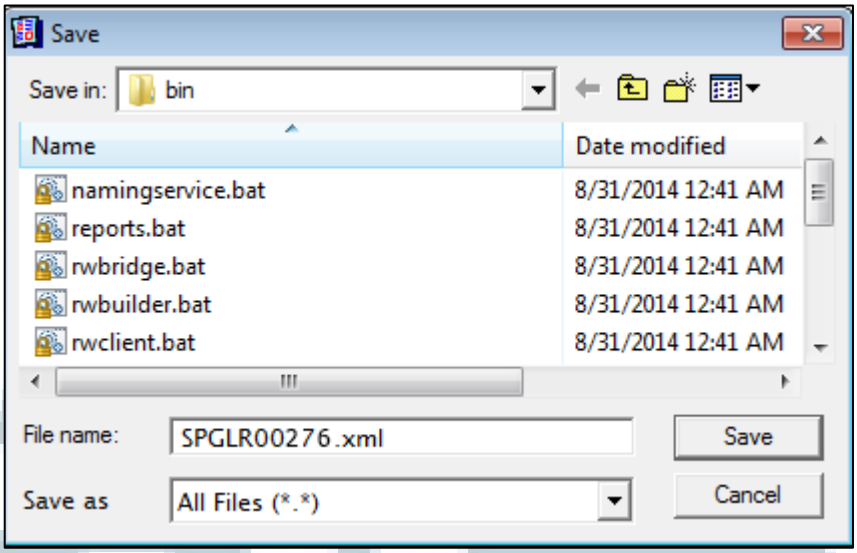


Gambar 3.162 – Menyimpan *report* dengan format *.rdf

d. Simpan data hasil *query* tersebut ke dalam *file* dengan tombol "*Generate to File*" pada *Oracle Report*. Format *file* yang digunakan adalah XML.

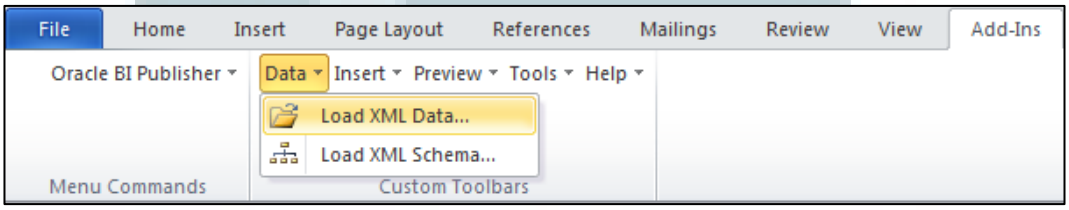


Gambar 3.163 – Meng-*generate report* ke *file* berbentuk XML

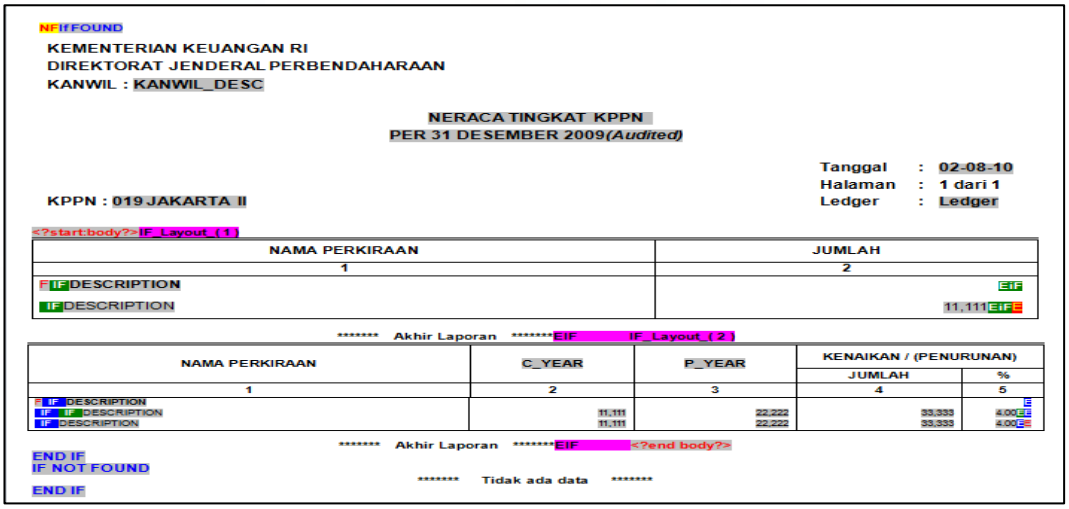


Gambar 3.164 – Menyimpan *report* dalam format *.xml

e. Data XML tersebut dimuat ke dalam *file report template* yang berupa *file* *.docx yang dimiliki oleh Kementrian Keuangan. Data tersebut dimuat dengan fungsi yang dimiliki *add-ins Oracle BI Publisher*.

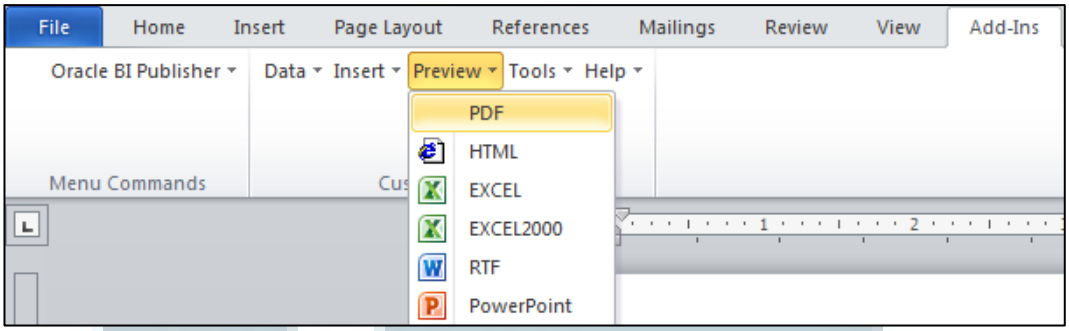


Gambar 3.165 – Memuat data xml ke *report template*



Gambar 3.166 – Format *report template* SPGLR00276

f. Tampilkan hasil *report* dengan menekan tombol *Preview*.



Gambar 3.167 – Menampilkan hasil *report* ke dalam PDF

g. Neraca Tingkat KPPN (SPGLR00276) telah selesai dibuat, namun demi kerahasiaan data, maka data (nilai angka) pada laporan ini tidak dapat ditampilkan.

NERACA TINGKAT KPPN
PER TANGGAL 31 MARET 2009

Ledger : Kas
Tanggal : 02-08-10
Halaman : 1

KPPN : 019 JAKARTA II

NAMA PERKIRAAN	JUMLAH
1	2
ASET	
ASET LANCAR	
Rekening Kas di KPPN	
Kas dalam Transito	
Kas di Bendahara Pengeluaran	
Kas pada Badan Layanan Umum	
JUMLAH ASET LANCAR	
JUMLAH ASET	
KEWAJIBAN	
KEWAJIBAN JANGKA PENDEK	
Utang Perhitungan Fihak Ketiga	
Utang kepada Pihak Ketiga	
JUMLAH KEWAJIBAN JANGKA PENDEK	
JUMLAH KEWAJIBAN	
EKUITAS DANA	
EKUITAS DANA LANCAR	
SAL	
SILPA	
Dana Lancar BLU	
JUMLAH EKUITAS DANA LANCAR	
JUMLAH EKUITAS DANA	
JUMLAH KEWAJIBAN DAN EKUITAS DANA	

Gambar 3.168 – Hasil *report* dalam bentuk PDF

3.3.2. Kendala dalam Pelaksanaan Magang

Selama periode kerja magang, peserta magang menemui beberapa kendala, sebagai berikut.

1. Spesifikasi *development server* yang disediakan oleh Kementrian Keuangan memiliki performa yang kurang sehingga proses *development* cukup terhambat.
2. Tidak adanya akses internet yang disediakan kepada tim *development* oleh Kementrian Keuangan, sehingga tim harus menggunakan akses internet pribadi.
3. *Data warehouse* dan konsep *Extract-Transform-Load* merupakan hal baru bagi peserta magang, dan peserta magang dituntut untuk dapat mempelajarinya dengan cepat.
4. Peserta magang tidak memiliki latar belakang di bidang akuntansi dan keuangan, sehingga harus mempelajarinya sendiri untuk kelancaraan pembuatan *data model* dan pengembangan *report*.

3.3.3. Solusi atas Kendala dalam Pelaksanaan Magang

Menghadapi kendala-kendala yang ditemukan, peserta magang melakukan beberapa solusi untuk mengurangi dampak kendala-kendala tersebut, sebagai berikut.

1. Melakukan konfigurasi dan optimasi *development server* dengan menonaktifkan fitur-fitur yang tidak diperlukan.

2. Menggunakan koneksi internet pribadi jika menemui masalah atau pertanyaan dalam *development*, dan juga mencari solusinya di rumah jika diperlukan.
3. Melakukan pembelajaraan pribadi baik dari media internet maupun meminta sarana dan prasarana dari perusahaan, yaitu berupa buku-buku pelatihan dan dokumentasi perusahaan. Tiga hal utama yang dipelajari peserta magang adalah *data warehouse*, PL/SQL serta akuntansi dan keuangan untuk mengembangkan *report*.