

## Extending the Banner Enterprise Data Warehouse

Presented by: Brian Large,  
Ellucian  
April 7<sup>th</sup>, 2014 @ 3:30pm  
Session ID 2617

---

---

---

---

---

---

---

---

### Session Objectives

- Describe Enterprise Data Warehouse (EDW) basics for Banner.
- Extend the EDW by bringing in an existing field from ODS.

---

---

---

---

---

---

---

---

## Understanding the EDW Architecture

---

---

---

---

---

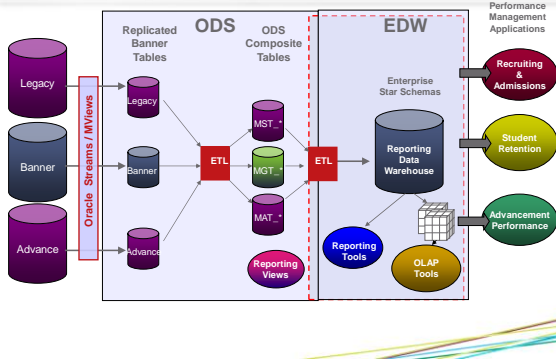
---

---

---

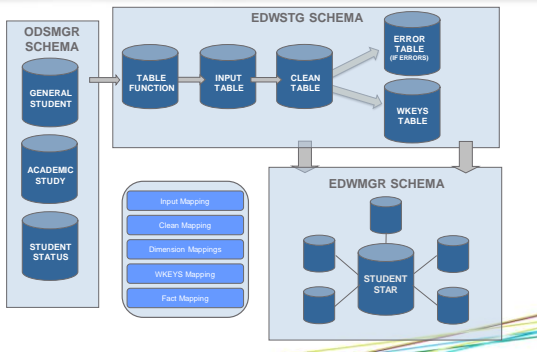
## Banner BI Architecture - EDW

ellucian  
LIVE  
2014



## EDW Baseline ETL Processing Flow

ellucian  
LIVE  
2014



## SGASTDN (SGBSTDN) Data

ellucian  
LIVE  
2014

As found in the EDW Baseline Analyze Student Progress Package

Student ID	SSN	Name	Student Status	Student Status Description	Student Population	Student Classification	Student Qualification	Student Qualification Description	Residency	Residency Description
200000	111111111	Jensen, Barbara	AL	Active	C	Continuing	SR	Senior	R	Resident

Student Summary

General Learner

New Terms: 2019 To Term: 2020

Student Status: 101 Continuing

Student Type: 101 Continuing

Residence: 101 Resident

Rate Assessment Rate: 101

Class: 101 Senior

Student Center Cycle: 101

Full or Part Time: 101 Full Time

Part Time: 101

Additional Information

From Term: 2000 To Term: 2020

SSN: 111111111

Residence: 101

Rate: 101

Citizenship: 101 U.S. Citizen

Curriculum Summary - Primary

Priority Term: 101

Program: 101

Catalog: 101

Student Type: 101

Campus: 101

College: 101

Degree: 101

Field of Study Summary

Priority Term: 101

Major: 101

Type: 101

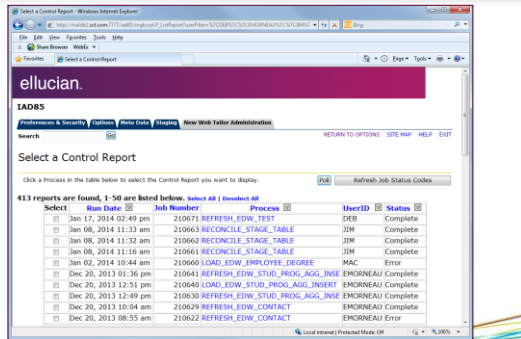
Field of Study: 101

Department: 101

Attached to Major: 101

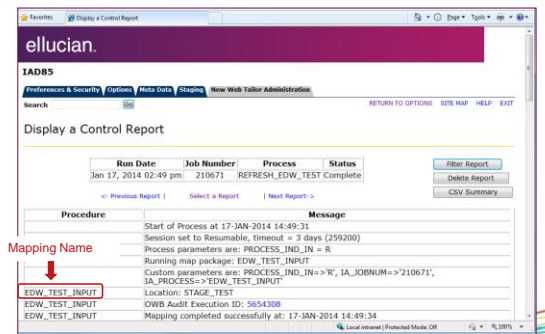
## Extending EDW – Control Reports

ellucian  
LIVE  
2014



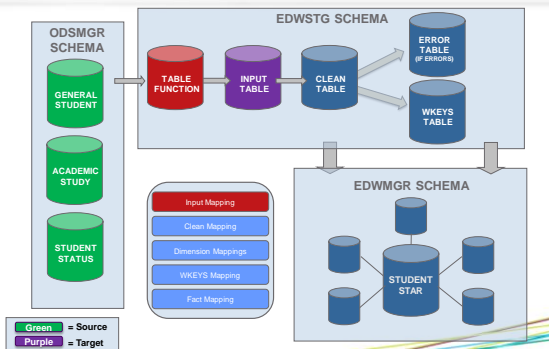
## Extending EDW – Typical ETL Job CR

ellucian  
LIVE  
2014



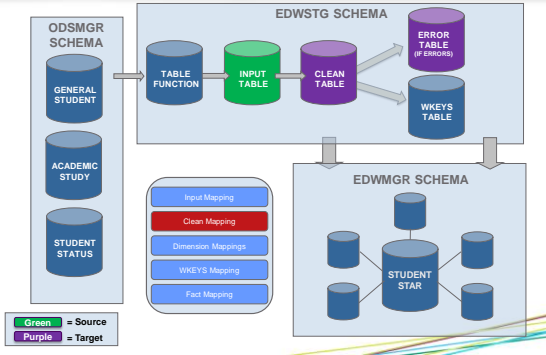
## EDW Baseline ETL Processing Flow

ellucian  
LIVE  
2014



## EDW Baseline ETL Processing Flow

ellucian  
LIVE  
2014



---

---

---

---

---

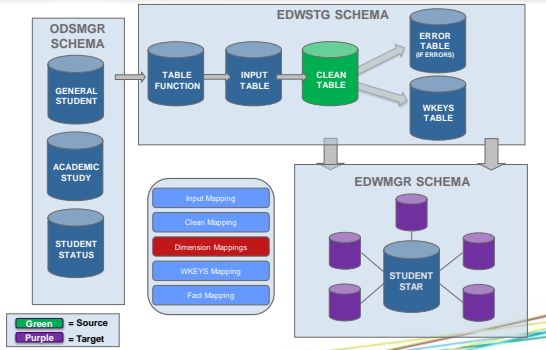
---

---

---

## EDW Baseline ETL Processing Flow

ellucian  
LIVE  
2014



---

---

---

---

---

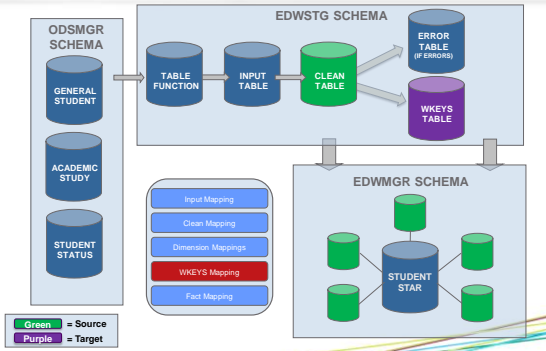
---

---

---

## EDW Baseline ETL Processing Flow

ellucian  
LIVE  
2014



---

---

---

---

---

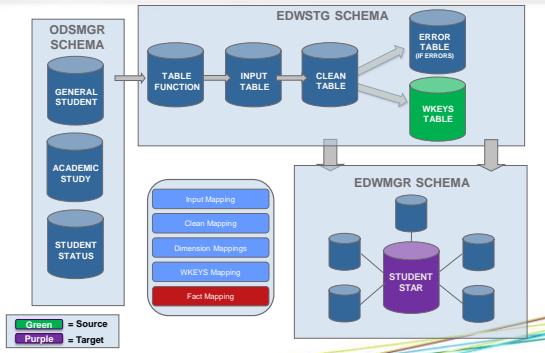
---

---

---

## EDW Baseline ETL Processing Flow

ellucian  
LIVE  
2014




---

---

---

---

---

---

---

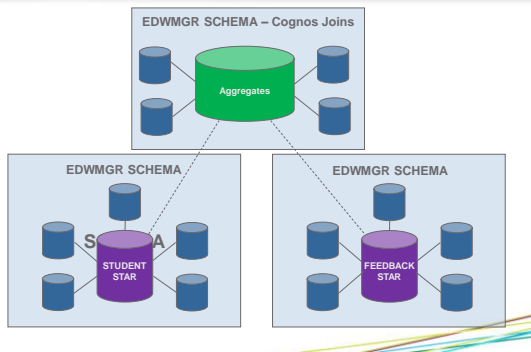
---

---

---

## EDW Aggregate Processing

ellucian  
LIVE  
2014




---

---

---

---

---

---

---

---

---

---

ellucian  
LIVE  
2014

Add existing ODS fields  
into an EDW Dimension

---

---

---

---

---

---

---

---

---

---

## Enterprise Data Warehouse Overview

ellucian  
LIVE  
2014

- Dimensionally modeled, uses Star Schemas.
- Fact tables.
  - Contain Measures for performing analysis
- Dimension Tables.
  - Contain Attributes.
  - Surrogate (Calculated) Keys.
  - Conformed.
- Every dimension and every fact table has five additional user defined fields for extensibility purposes.



---

---

---

---

---

---

---

## Considerations when Extending Models

ellucian  
LIVE  
2014

- Where/how is that data captured?
  - This tells you what the source is:
    - Is it already in the ODS or some other system?
    - Will it be automatically incorporated into the refresh?
- How will the new object be used – as an attribute or as a measure?
  - This tells you whether it should be added to a dimension or fact table.
- If a measure, is it captured at the same level of granularity as the other data in the star?
  - If different, this tells you what adjustments to make on either the reporting or ETL side to accommodate the difference.



---

---

---

---

---

---

---

## Adding a New Value using the USER\_DEFINED Fields

ellucian  
LIVE  
2014

**Scenario:** You are asked to bring the ODS field **TAXONOMY\_OF\_PROGRAM** into the EDW for reporting on courses.



---

---

---

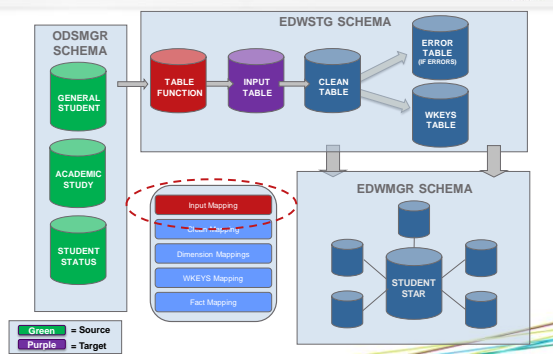
---

---

---

---

ellucian  
LIVE  
2014

 $f(x)$ 

ellucian  
LIVE  
2014

Run

Query

```
select * from all_tables where column_name like 'TABLES%';
```

Query Result 1

OWNER	TABLE_NAME	COLUMN_NAME	DATA_TYPE	DATA_TYPE_MOD	DATA_TYPE_SCALE
SYS	ALL_TABLES	TABLE_NAME	VARCHAR2		
SYS	ALL_TABLES	COLUMN_NAME	VARCHAR2		
SYS	ALL_TABLES	DATA_TYPE	VARCHAR2		
SYS	ALL_TABLES	DATA_TYPE_MOD	VARCHAR2		
SYS	ALL_TABLES	DATA_TYPE_SCALE	VARCHAR2		
SYS	ALL_TABLES	TABLESPACE	VARCHAR2		
SYS	ALL_TABLES	PARTITION_NAME	VARCHAR2		
SYS	ALL_TABLES	PARTITION_POSITION	VARCHAR2		
SYS	ALL_TABLES	TABLESPACE	VARCHAR2		
SYS	ALL_TABLES	TABLESPACE	VARCHAR2		

ellucian  
LIVE  
2014

```

*
*
Load/Refresh Combined Admissions Source Star
Load/Refresh Contact Star
Load/Refresh Course Instructor Star
Load/Refresh Course Section Star
Load/Refresh Course Meeting Time Star
Load/Refresh EM Campaign EDW Star
*
*
Load/Refresh Student Attribute Star
Load/Refresh Student Cohort Star
Load/Refresh Student Course Star
Load/Refresh Student Course Attribute Star
Load/Refresh Student Course Grade Change Star
Load/Refresh Student Star
*
*

```

ellucian  
S LIVE  
2014

- | File          | Line | Code                           | Commit          |
|---------------|------|--------------------------------|-----------------|
| src/asm/asm.h | 1    | #ifndef ASM_H                  | 1587-08-20-2015 |
| src/asm/asm.h | 2    | #define ASM_H                  | 1587-08-20-2015 |
| src/asm/asm.h | 3    | #include "asm_constants.h"     | 1587-08-20-2015 |
| src/asm/asm.h | 4    | #include "asm_registers.h"     | 1587-08-20-2015 |
| src/asm/asm.h | 5    | #include "asm_instructions.h"  | 1587-08-20-2015 |
| src/asm/asm.h | 6    | #include "asm_exceptions.h"    | 1587-08-20-2015 |
| src/asm/asm.h | 7    | #include "asm_memory.h"        | 1587-08-20-2015 |
| src/asm/asm.h | 8    | #include "asm_io.h"            | 1587-08-20-2015 |
| src/asm/asm.h | 9    | #include "asm_interrupts.h"    | 1587-08-20-2015 |
| src/asm/asm.h | 10   | #include "asm_timer.h"         | 1587-08-20-2015 |
| src/asm/asm.h | 11   | #include "asm_debug.h"         | 1587-08-20-2015 |
| src/asm/asm.h | 12   | #include "asm_test.h"          | 1587-08-20-2015 |
| src/asm/asm.h | 13   | #include "asm_util.h"          | 1587-08-20-2015 |
| src/asm/asm.h | 14   | #include "asm_error.h"         | 1587-08-20-2015 |
| src/asm/asm.h | 15   | #include "asm_stack.h"         | 1587-08-20-2015 |
| src/asm/asm.h | 16   | #include "asm_memory_map.h"    | 1587-08-20-2015 |
| src/asm/asm.h | 17   | #include "asm_interrupt_map.h" | 1587-08-20-2015 |
| src/asm/asm.h | 18   | #include "asm_timer_map.h"     | 1587-08-20-2015 |
| src/asm/asm.h | 19   | #include "asm_debug_map.h"     | 1587-08-20-2015 |
| src/asm/asm.h | 20   | #include "asm_test_map.h"      | 1587-08-20-2015 |
| src/asm/asm.h | 21   | #include "asm_util_map.h"      | 1587-08-20-2015 |
| src/asm/asm.h | 22   | #include "asm_error_map.h"     | 1587-08-20-2015 |
| src/asm/asm.h | 23   | #include "asm_stack_map.h"     | 1587-08-20-2015 |
| src/asm/asm.h | 24   | #include "asm_memory_map.h"    | 1587-08-20-2015 |
| src/asm/asm.h | 25   | #include "asm_interrupt_map.h" | 1587-08-20-2015 |
| src/asm/asm.h | 26   | #include "asm_timer_map.h"     | 1587-08-20-2015 |
| src/asm/asm.h | 27   | #include "asm_debug_map.h"     | 1587-08-20-2015 |
| src/asm/asm.h | 28   | #include "asm_test_map.h"      | 1587-08-20-2015 |
| src/asm/asm.h | 29   | #include "asm_util_map.h"      | 1587-08-20-2015 |
| src/asm/asm.h | 30   | #include "asm_error_map.h"     | 1587-08-20-2015 |
| src/asm/asm.h | 31   | #include "asm_stack_map.h"     | 1587-08-20-2015 |
| src/asm/asm.h | 32   | #include "asm_memory_map.h"    | 1587-08-20-2015 |
| src/asm/asm.h | 33   | #include "asm_interrupt_map.h" | 1587-08-20-2015 |
| src/asm/asm.h | 34   | #include "asm_timer_map.h"     | 1587-08-20-2015 |
| src/asm/asm.h | 35   | #include "asm_debug_map.h"     | 1587-08-20-2015 |
| src/asm/asm.h | 36   | #include "asm_test_map.h"      | 1587-08-20-2015 |
| src/asm/asm.h | 37   | #include "asm_util_map.h"      | 1587-08-20-2015 |
| src/asm/asm.h | 38   | #include "asm_error_map.h"     | 1587-08-20-2015 |
| src/asm/asm.h | 39   | #include "asm_stack_map.h"     | 1587-08-20-2015 |
| src/asm/asm.h | 40   | #include "asm_memory_map.h"    | 1587-08-20-2015 |
| src/asm/asm.h | 41   | #include "asm_interrupt_map.h" | 1587-08-20-2015 |
| src/asm/asm.h | 42   | #include "asm_timer_map.h"     | 1587-08-20-2015 |
| src/asm/asm.h | 43   | #include "asm_debug_map.h"     | 1587-08-20-2015 |
| src/asm/asm.h | 44   | #include "asm_test_map.h"      | 1587-08-20-2015 |
| src/asm/asm.h | 45   | #include "asm_util_map.h"      | 1587-08-20-2015 |
| src/asm/asm.h | 46   | #include "asm_error_map.h"     | 1587-08-20-2015 |
| src/asm/asm.h | 47   | #include "asm_stack_map.h"     | 1587-08-20-2015 |
| src/asm/asm.h | 48   | #include "asm_memory_map.h"    | 1587-08-20-2015 |
| src/asm/asm.h | 49   | #include "asm_interrupt_map.h" | 1587-08-20-2015 |
| src/asm/asm.h | 50   | #include "asm_timer_map.h"     | 1587-08-20-2015 |
| src/asm/asm.h | 51   | #include "asm_debug_map.h"     | 1587-08-20-2015 |
| src/asm/asm.h | 52   | #include "asm_test_map.h"      | 1587-08-20-2015 |
| src/asm/asm.h | 53   | #include "asm_util_map.h"      | 1587-08-20-2015 |
| src/asm/asm.h | 54   | #include "asm_error_map.h"     | 1587-08-20-2015 |
| src/asm/asm.h | 55   | #include "asm_stack_map.h"     | 1587-08-20-2015 |
| src/asm/asm.h | 56   | #include "asm_memory_map.h"    | 1587-08-20-2015 |
| src/asm/asm.h | 57   | #include "asm_interrupt_map.h" | 1587-08-20-2015 |
| src/asm/asm.h | 58   | #include "asm_timer_map.h"     | 1587-08-20-2015 |
| src/asm/asm.h | 59   | #include "asm_debug_map.h"     | 1587-08-20-2015 |
| src/asm/asm.h | 60   | #include "asm_test_map.h"      | 1587-08-20-2015 |
| src/asm/asm.h | 61   | #include "asm_util_map.h"      | 1587-08-20-2015 |
| src/asm/asm.h | 62   | #include "asm_error_map.h"     | 1587-08-20-2015 |
| src/asm/asm.h | 63   | #include "asm_stack_map.h"     | 1587-08-20-2015 |
| src/asm/asm.h | 64   | #include "asm_memory_map.h"    | 1587-08-20-2015 |
| src/asm/asm.h | 65   | #include "asm_interrupt_map.h" | 1587-08-20-2015 |
| src/asm/asm.h | 66   | #include "asm_timer_map.h"     | 1587-08-20-2015 |
| src/asm/asm.h | 67   | #include "asm_debug_map.h"     | 1587-08-20-2015 |
| src/asm/asm.h | 68   | #include "asm_test_map.h"      | 1587-08-20-2015 |

ellucian  
LIVE  
2014

- [illegible]

SQL finds Fact tables associated with the Dimension via referential integrity



## USER\_DEFINED Fields

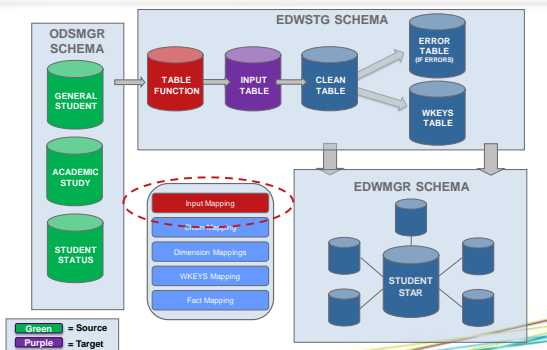
ellucian  
LIVE  
2014

- Use the USER\_ATTRIBUTE\_01 column in the WDT\_COURSE\_SECTION table to store TAXONOMY\_OF\_PROGRAM (TOPS) data.

<b>Banner</b>	SCBSUPP.SCBSUPP_TOPS_CODE
<b>ODS View</b>	COURSE_CATALOG.TAXONOMY_OF_PROGRAM_CODE
<b>ODS Table</b>	MST_COURSE_SUPPLEMENTAL. TAXONOMY_OF_PROGRAM_CODE
<b>EDW Star</b>	COURSE_SECTION
<b>EDW Dimension</b>	WDT_COURSE_SECTION.USER_ATTRIBUTE_01

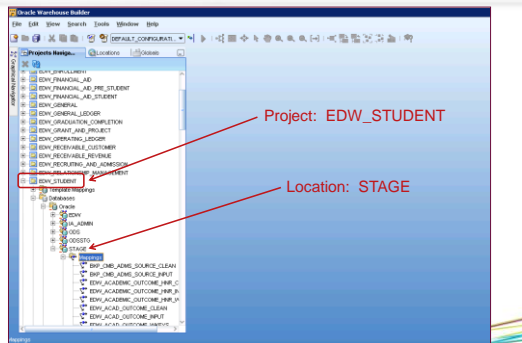
## EDW Baseline ETL Processing Flow

ellucian  
LIVE  
2014



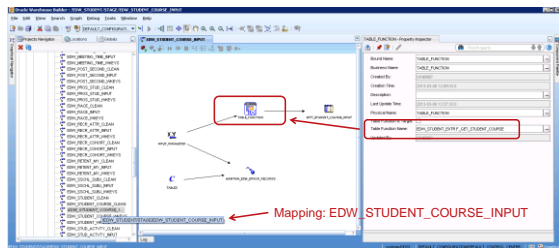
## Oracle Warehouse Builder (OWB)

ellucian  
LIVE  
2014



## INPUT Mapping Flow

ellucian  
LIVE  
2014



## Table Functions

ellucian  
LIVE  
2014

- Use table functions to extract the data from the ODS to the EDW for loads/refreshes.
- There is one table function per star schema.
- Snapshot stars' table functions live in individual PL/SQL packages.
- Operational stars' table functions live in one package per product.
- Table function packages are created in the EDWSTG schema.
- Package naming convention is EDW\_<product or snapshot star>\_EXTR.

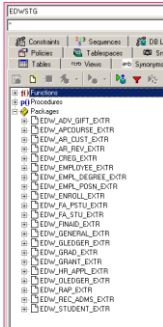
## Table Functions

ellucian  
LIVE  
2014

- Pipelined table function.
- Source for INPUT mapping.
- Contains *driver* SQL to return rows.
- Uses function-specific and *shared* cursors to access ODS data.
- Contains USER\_DEFINED fields for extensibility.
- Testable from SQL using the TABLE keyword:  
**SELECT \* FROM TABLE( f\_getData() );**

## Table Function Packages

ellucian  
LIVE  
2014



---

---

---

---

---

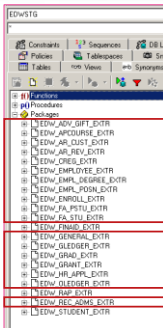
---

---

---

## Table Function Packages

ellucian  
LIVE  
2014



Snapshot Stars

One Package Per Star



---

---

---

---

---

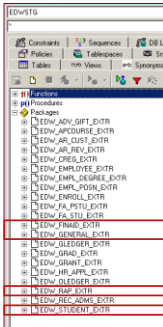
---

---

---

## Table Function Packages

ellucian  
LIVE  
2014



Operational Stars

One Package Per Product



---

---

---

---

---

---

---

---

# Table Function Recommendations

ellucian  
LIVE  
2014

- Copy the table function package script to an institution-specific directory outside the baseline EDW code tree.
- Rename the script to reflect it is modified.
- Do not change the name of the object created by the script.
- Update the script to include desired attributes or measures.
- Any patches or upgrades to the baseline version of these objects must be merged with the modified code.



---

---

---

---

---

---

---

# Table Function Structure

ellucian  
LIVE  
2014

<b>Driving Cursor</b>	Selects the population drives the extract.
<b>Supporting Cursors</b>	Selects support data related to driving population.
<b>Return Row Definition</b>	Where selected values are assigned to staging input table columns.



---

---

---

---

---

---

---

# Modifying a Table Function Cursor

ellucian  
LIVE  
2014

- In our example, we add **Taxonomy\_Of\_Program** to a user defined field in the Course Section star, in the **edw\_student\_extr.f\_get\_course\_section** table function.
- Taxonomy data is already fetched as part of a supporting cursor, selected from the ODS composite table **MST\_COURSE\_SUPPLEMENTAL**.



---

---

---

---

---

---

---

## WDT\_COURSE\_SECTION

ellucian  
LIVE  
2014

```
WDT_COURSE_SECTION
COURSE_SECTION_KEY: NUMBER
COURSE_SECTION: VARCHAR2(63)
COURSE_SECTION_SD: VARCHAR2(255)
COURSE_SECTION_LD: VARCHAR2(255)
INSTRUCTION_DELIVERY_MODE: VARCHAR2(63)
INSTRUCTION_DELIVERY_MODE_SD: VARCHAR2(255)
INSTRUCTION_DELIVERY_MODE_LD: VARCHAR2(255)
GRADABLE_IND: NUMBER
GRADABLE_IND_SD: VARCHAR2(255)
GRADABLE_IND_LD: VARCHAR2(255)
SECTION_CROSS_LIST: VARCHAR2(63)
SECTION_CROSS_LIST_SD: VARCHAR2(255)
SECTION_CROSS_LIST_LD: VARCHAR2(255)
USER_ATTRIBUTE_01: VARCHAR2(63)
USER_ATTRIBUTE_01_SD: VARCHAR2(255)
USER_ATTRIBUTE_01_LD: VARCHAR2(255)
USER_ATTRIBUTE_02: VARCHAR2(63)
USER_ATTRIBUTE_02_SD: VARCHAR2(255)
USER_ATTRIBUTE_02_LD: VARCHAR2(255)
USER_ATTRIBUTE_03: VARCHAR2(63)
USER_ATTRIBUTE_03_SD: VARCHAR2(255)
USER_ATTRIBUTE_03_LD: VARCHAR2(255)
USER_ATTRIBUTE_04: VARCHAR2(63)
USER_ATTRIBUTE_04_SD: VARCHAR2(255)
USER_ATTRIBUTE_04_LD: VARCHAR2(255)
USER_ATTRIBUTE_05: VARCHAR2(63)
USER_ATTRIBUTE_05_SD: VARCHAR2(255)
USER_ATTRIBUTE_05_LD: VARCHAR2(255)
SYSTEM_LOAD_PROCESS: VARCHAR2(30)
SYSTEM_LOAD_TIMESTAMP: DATE
```

## f\_get\_course\_section() Cursor Logic

ellucian  
LIVE  
2014

```
CURSOR get_course_suppl(def_multi_source in VARCHAR2,multi_src in VARCHAR2,
subject in VARCHAR2, course_number in VARCHAR2,academic_period in VARCHAR2) IS
SELECT *
FROM WDT_COURSE_SUPPLEMENTAL
WHERE SUBJECT = subject_in
AND COURSE_NUMBER = course_number_in
AND academic_period in BETWEEN ACADEMIC_PERIOD_START AND
ACADEMIC_PERIOD_END
AND NVL(MTF_VALUE, def_multi_source_in) = multi_src_in;

...
function f_get_course_section(academic_year in VARCHAR2, process_ind in VARCHAR2,
process_date DATE, debug_var in VARCHAR2 DEFAULT NULL) RETURN
tab_student_course_extr PIPELINED IS
--
course_suppl_rec EDW_STUDENT_EXTR.GET_COURSE_SUPPL%ROWTYPE;
...
OPEN get_course_suppl(default_rec.multi_source_cleanse_value,
student_course_info_recs(i).multi_source, student_course_info_recs(i).subject,
student_course_info_recs(i).course_number,
student_course_info_recs(i).academic_period);
FETCH get_course_suppl INTO course_suppl_rec;
CLOSE get_course_suppl;
```

## f\_get\_course\_section() Cursor Logic

ellucian  
LIVE  
2014

### BASELINE:

```
-- Course Section
ret_row.course_section := crse_sec_rec.course_section;
ret_row.instruction_delivery_mode := crse_sec_rec.instruction_delivery_mode;
ret_row.gradable_ind := case when crse_sec_rec.gradable_ind = 'Y' then 1 else 0
end;
ret_row.section_cross_list := crse_sec_rec.section_cross_list;
ret_row.crsn_user_attribute_01 := NULL;
ret_row.crsn_user_attribute_02 := NULL;
ret_row.crsn_user_attribute_03 := NULL;
ret_row.crsn_user_attribute_04 := NULL;
ret_row.crsn_user_attribute_05 := NULL;
```

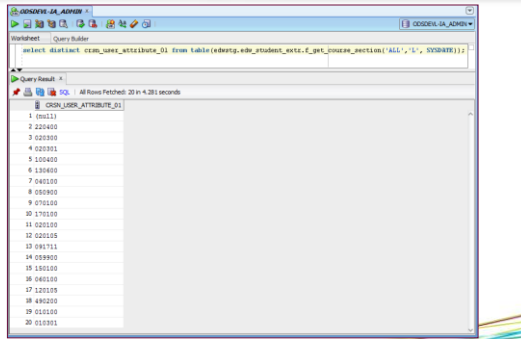
### MODIFIED:

```
-- Course Section
ret_row.course_section := crse_sec_rec.course_section;
ret_row.instruction_delivery_mode := crse_sec_rec.instruction_delivery_mode;
ret_row.gradable_ind := case when crse_sec_rec.gradable_ind = 'Y' then 1 else 0 end;
ret_row.section_cross_list := crse_sec_rec.section_cross_list;
ret_row.crsn_user_attribute_01 := course_suppl_rec.TAXONOMY_OF_PROGRAM;
ret_row.crsn_user_attribute_02 := NULL;
ret_row.crsn_user_attribute_03 := NULL;
ret_row.crsn_user_attribute_04 := NULL;
ret_row.crsn_user_attribute_05 := NULL;
```

## Table Function Compilation

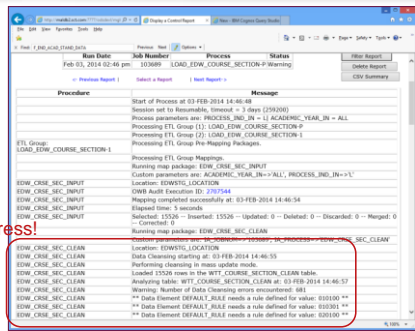
- Compile the modified package as EDWSTG.
- Grant execute on the package to IA\_ADMIN.
  - This ensures permissions are not lost.
- No Updates to OWB mappings are needed.
- Testable from SQL query or Admin UI job.

## Test from SQL Query



## Test from Admin UI Job - After

Progress!



## Step 2: Cleansing Rules

ellucian  
LIVE  
2014



---

---

---

---

---

---

---

## Cleansing Process Review

ellucian  
LIVE  
2014

- Data cleansing is the process of verifying source system code values and translating them to standardized code values *and* descriptions in the warehouse.
- Configure data cleansing in Admin UI.
  - Options -> Set Up & Maintain Cleansing Parameters
  - Options -> Set Up Parameters

---

---

---

---

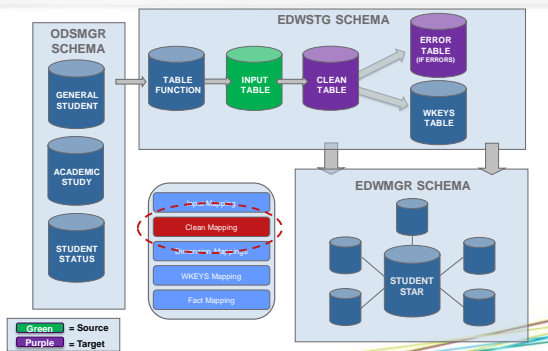
---

---

---

## EDW Baseline ETL Processing Flow

ellucian  
LIVE  
2014



---

---

---

---

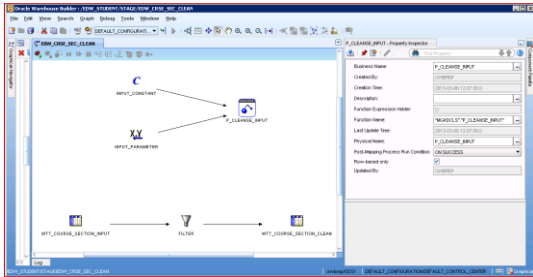
---

---

---

## CLEAN Mapping Flow

ellucian  
LIVE  
2014



## Cleansing Rule Configuration

ellucian  
LIVE  
2014

- **Cleansing Rule**
  - Identifies the source query used to generate default cleansing values and translations for data element.
  - Stored in IA\_ADMIN.MGRCRUL.
- **Cleansing Data Element**
  - Links cleansing rule to the dimension and column to cleanse.
  - Stored in IA\_ADMIN.MGRCDIM.
- **Cleansing Data Element Parameter**
  - Links cleansing rule to the star, dimension, and input table field to cleanse.
  - Stored in IA\_ADMIN.MTVPARM.
    - MTVPARM\_INTERNAL\_GROUP = CLEANSING DATA ELEMENT

## Example Cleansing Rule Configuration

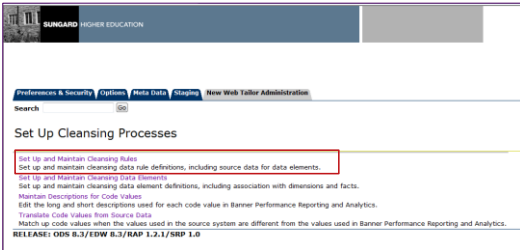
ellucian  
LIVE  
2014

- **Cleansing Rule**
  - Example Source table: MST\_COURSE\_SUPPLEMENTAL
  - Example Data Element: TAXONOMY\_CODE Create
- **Cleansing Data Element**
  - Example Data Element: TAXONOMY\_CODE
  - Example Dimension: WDT\_COURSE\_SECTION
  - Example Dim Column: USER\_ATTRIBUTE\_01 Update
- **Cleansing Data Element Parameter**
  - Example Dimension: TAXONOMY\_CODE
  - Example Dim Column: WDT\_COURSE\_SECTION
  - Example Star: USER\_ATTRIBUTE\_01
  - Example Table Field Col: CRSN\_USER\_ATTRIBUTE\_01 No Change



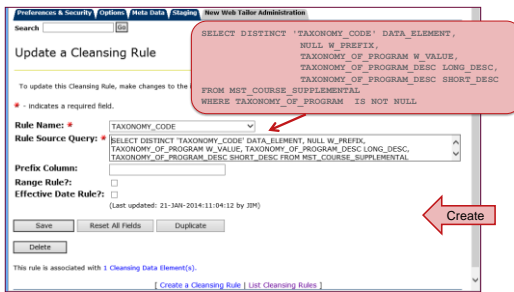
## Cleansing Rule Navigation

ellucian  
LIVE  
2014



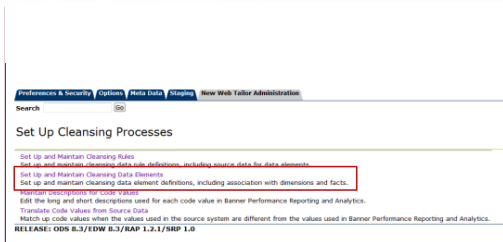
## Cleansing Rule Definition

ellucian  
LIVE  
2014



## Cleansing Data Element Definition

ellucian  
LIVE  
2014



## Cleansing Data Element Definition

ellucian  
LIVE  
2014

Select a Cleansing Data Element

Click a Column in the table below to select the Cleansing Data Element you want to update or delete, or change the search criteria and click Search.

Dimension	Column	Def Null Value	Def Null Short Desc	Def Null Long Desc	Cleansing Ind?	Cleansing Rule	Sys Req?
WDT_COURSE_SECTION	COURSE_SECTION				Y (Disable)	COURSE_SECTION	Y
WDT_COURSE_SECTION	GRADABLE_IND				Y (Disable)	GRADABLE_IND	Y
WDT_COURSE_SECTION	INSTRUCTION_DELIVERY_MODE				Y (Disable)	INSTRUCTION_DELIVERY_MODE	Y
WDT_COURSE_SECTION	SECTION_CROSS_LIST				Y (Disable)	SECTION_CROSS_LIST	Y
WDT_COURSE_SECTION	USER_ATTRIBUTE_01				Y (Disable)	TAXONOMY_CODE	Y
WDT_COURSE_SECTION	USER_ATTRIBUTE_02				N (Enable)	DEFAULT_RULE	Y
WDT_COURSE_SECTION	USER_ATTRIBUTE_03				N (Enable)	DEFAULT_RULE	Y
WDT_COURSE_SECTION	USER_ATTRIBUTE_04				N (Enable)	DEFAULT_RULE	Y
WDT_COURSE_SECTION	USER_ATTRIBUTE_05				N (Enable)	DEFAULT_RULE	Y

1 Create a Cleansing Rule 1 List Cleansing Rules 1

Update

## Cleansing Data Element Definition

ellucian  
LIVE  
2014

Click a Description in the table below to select the Parameter you want to update or delete, or change the search criteria and click Search.

Internal Group	Internal Code 1	Internal Code 2	Internal Code Sequence	External Code	Description
CLEANSING COURSE_SECTION WDT_COURSE_SECTION DATA ELEMENTS			1	COURSE_SECTION	COURSE_SECTION
CLEANSING COURSE_SECTION WDT_COURSE_SECTION DATA ELEMENTS			1	CRSN_USER_ATTRIBUTE_01	USER_ATTRIBUTE_01
CLEANSING COURSE_SECTION WDT_COURSE_SECTION DATA ELEMENTS			1	CRSN_USER_ATTRIBUTE_02	USER_ATTRIBUTE_02
CLEANSING COURSE_SECTION WDT_COURSE_SECTION DATA ELEMENTS			1	CRSN_USER_ATTRIBUTE_03	USER_ATTRIBUTE_03
CLEANSING COURSE_SECTION WDT_COURSE_SECTION DATA ELEMENTS			1	CRSN_USER_ATTRIBUTE_04	USER_ATTRIBUTE_04
CLEANSING COURSE_SECTION WDT_COURSE_SECTION DATA ELEMENTS			1	CRSN_USER_ATTRIBUTE_05	USER_ATTRIBUTE_05
CLEANSING COURSE_SECTION WDT_COURSE_SECTION DATA ELEMENTS			1	GRADABLE_IND	GRADABLE_IND
CLEANSING COURSE_SECTION WDT_COURSE_SECTION DATA ELEMENTS			1	INSTRUCTION_DELIVERY_MODE	INSTRUCTION_DELIVERY_MODE

4

No Change

## Step 3: Metadata Creation

ellucian  
LIVE  
2014

metadata  
describe  
one  
information  
Data

## Updating Metadata

ellucian  
LIVE  
2014

- Administrative User Interface
  - Add the USER\_ATTRIBUTE\_01 column and provide a meaningful Business Name/Definition.
  - Publish the Metadata report.
- Reporting Tool (Cognos)
  - Framework Manager
    - Rename USER\_ATTRIBUTE\_01 to a new column name (TAXONOMY\_OF\_PROGRAM).
    - Publish the package.

## Updating Metadata (Maintain)

ellucian  
LIVE  
2014

## Updating Metadata (Publish)

ellucian  
LIVE  
2014

Target Column	Business Definition	Database Data Type	Source Name	Source Column	Local Dimension Target	Star Name
COURSE_SECTION_ID	Key for the course section dimension	NUMBER(10,0)	WDT_COURSE_SECTION	COURSE_SECTION_ID	NO	COURSE_SECTION
COURSE_SECTION_NAME	Course section name used with the academic period subject and course number to uniquely identify a section offering. An offering number can only be used once in a term. A section number is used in an academic period. However, multiple offerings of a course can share a same course number number.	VARCHAR2(255)	SCHEDULE_OFFERING	OFFERING_NUMBER	NO	COURSE_SECTION
COURSE_SECTION_ID	Key for the course section dimension	NUMBER(10,0)	WDT_COURSE_SECTION	COURSE_SECTION_ID	NO	COURSE_SECTION
COURSE_SECTION_ID	Key for the course section dimension	NUMBER(10,0)	WDT_COURSE_SECTION	COURSE_SECTION_ID	NO	COURSE_SECTION
INSTRUCTIONAL_METHOD_ID	Instructional method used with the academic period subject and course number to uniquely identify a section offering. An offering number can only be used once in a term. A section number is used in an academic period. However, multiple offerings of a course can share a same course number number.	VARCHAR2(255)	SCHEDULE_OFFERING	INSTRUCTIONAL_METHOD_ID	NO	COURSE_SECTION
INSTRUCTIONAL_METHOD_ID	Instructional method used with the academic period subject and course number to uniquely identify a section offering. An offering number can only be used once in a term. A section number is used in an academic period. However, multiple offerings of a course can share a same course number number.	VARCHAR2(255)	SCHEDULE_OFFERING	INSTRUCTIONAL_METHOD_ID	NO	COURSE_SECTION
GRADABLE_IND	Indicates whether the course section can be graded	NUMBER	SCHEDULE_OFFERING	GRADABLE_IND	NO	COURSE_SECTION
GRADABLE_IND	Indicates whether the course section can be graded	NUMBER	SCHEDULE_OFFERING	GRADABLE_IND	NO	COURSE_SECTION
GRADABLE_IND	Indicates whether the course section can be graded	NUMBER	SCHEDULE_OFFERING	GRADABLE_IND	NO	COURSE_SECTION
SECTION_CROSS_LIST_ID	Key for the section cross list dimension	NUMBER(10,0)	WDT_SECTION_CROSS_LIST	SECTION_CROSS_LIST_ID	NO	COURSE_SECTION
SECTION_CROSS_LIST_ID	Key for the section cross list dimension	NUMBER(10,0)	WDT_SECTION_CROSS_LIST	SECTION_CROSS_LIST_ID	NO	COURSE_SECTION
SECTION_CROSS_LIST_ID	Key for the section cross list dimension	NUMBER(10,0)	WDT_SECTION_CROSS_LIST	SECTION_CROSS_LIST_ID	NO	COURSE_SECTION
SYSTEM_LOAD_PRIORITY	Value of the system load priority that created the star during the last active process	NUMBER(10,0)	WDT_SYSTEM_LOAD_PRIORITY	SYSTEM_LOAD_PRIORITY	NO	COURSE_SECTION
USER_ATTRIBUTE_01	Taxonomy of Program data	VARCHAR2(255)	WDT_USER_ATTRIBUTE_01	USER_ATTRIBUTE_01	YES	COURSE_SECTION

## Step 4: Cognos Modification

ellucian  
LIVE  
2014



---

---

---

---

---

---

---

## Cognos Metadata Modification

ellucian  
LIVE  
2014

- Update reporting metadata to include the new column.
- User Attribute and Measure fields exist in the database layer.
- Rename WDT\_ACADEMIC\_TIME's attributes USER\_ATTRIBUTE\_01, USER\_ATTRIBUTE\_SD, and USER\_ATTRIBUTE\_LD fields to business names.
  - For example: Academic Period Type, Academic Period Type Description, and Academic Period Type Long Description.



---

---

---

---

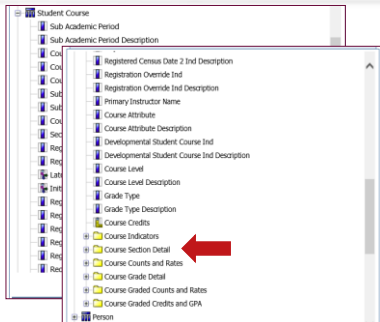
---

---

---

## Query Subjects in Cognos

ellucian  
LIVE  
2014



---

---

---

---

---

---

---

## Retracing Our Steps ...

ellucian  
LIVE  
2014

- Step 1 Table Function Modification
  - Change and/or enhance data being loaded from ODS
- Step 2 Cleansing Rule Creation
  - Ensure valid data is loaded into the EDW
- Step 3 Meta Data Creation
  - Update business definitions of data elements
- Step 4 Reporting Metadata Modification
  - Make new/updated columns available for end users



---

---

---

---

---

---

---

ellucian  
LIVE  
2014

Thank You!

Brian Large  
brian.large@ellucian.com

Please complete the online session evaluation form  
Session ID 2617

© 2014 Ellucian. All rights reserved.



---

---

---

---

---

---

---