Lecture 6

More ETL, Data Marts, Elaborate HW Assignment

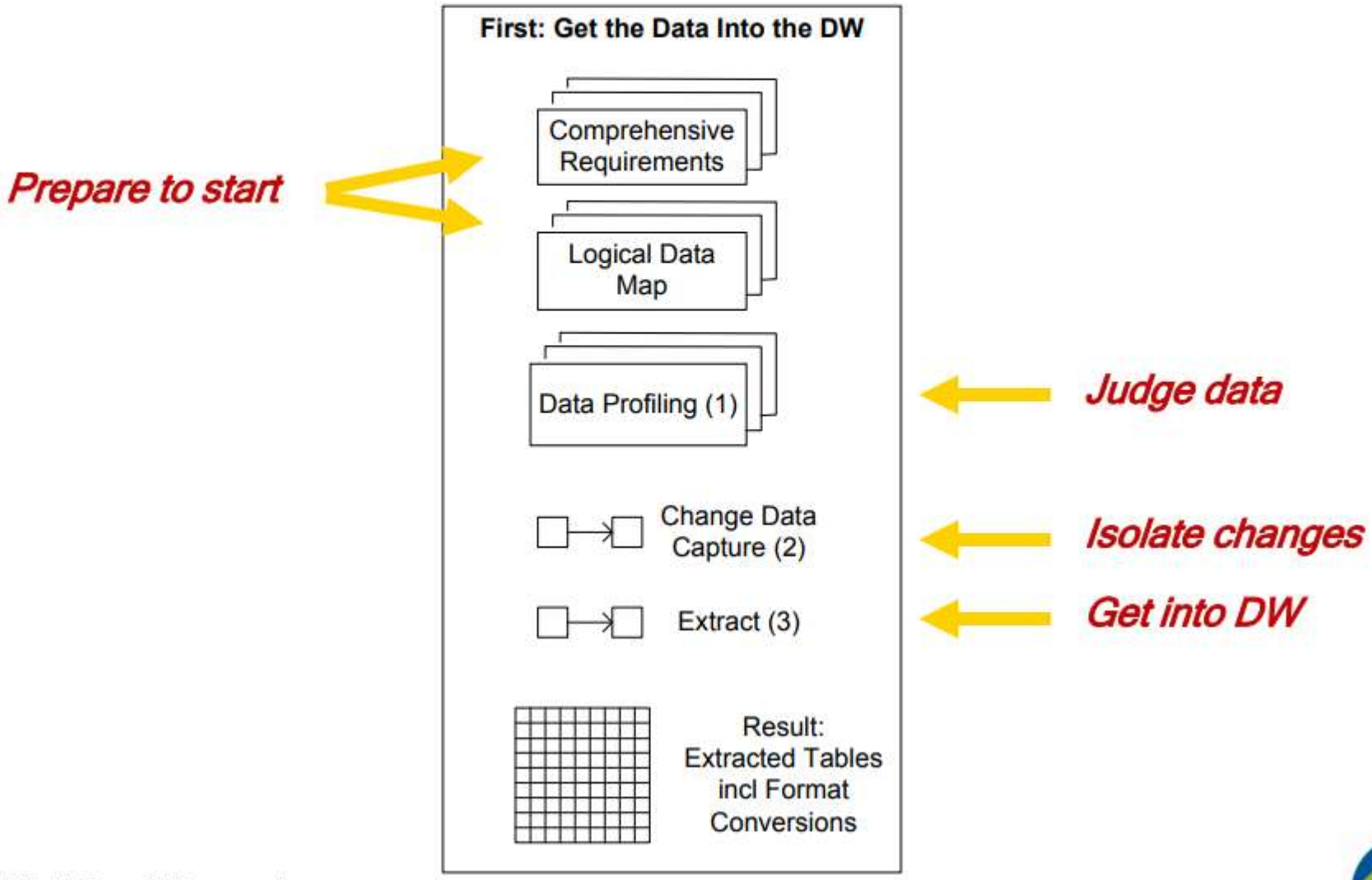
Breitzman 10/9/17

34 ETL Subsystems

- According to Ralph Kimball, there are 34 subsystems related to the ETL process. (Actually his process is ETLM – Extract, Transform, Load, Manage)
- Three subsystems focus on extracting data from source systems.
- Five subsystems deal with value-added cleaning and conforming, including dimensional structures to monitor quality errors.
- Thirteen subsystems deliver data as dimensional structures to the final BI layer, such as a subsystem to implement slowly changing dimension techniques.
- Thirteen subsystems help manage the production ETL environment.
- Don’t worry, we’re not going to talk about all of them tonight, but we’ll talk about the ones we need for the next phase of our grocery data warehouse.
- Next 4 slides are ‘borrowed’ from Kimball group website

E: Getting the Data Into the DW

Note: Numbers in the parentheses refer to Kimball's 34 ETL subsystems.



KIMBALL GROUP

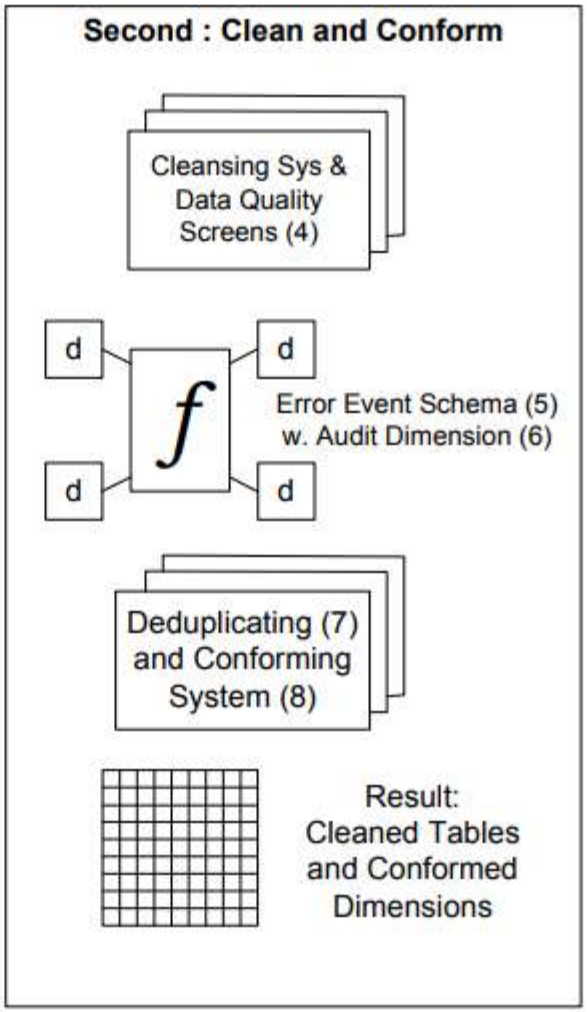
T: Clean and Conform

Note: Numbers in the parentheses refer to Kimball's 34 ETL subsystems.

Cleaning machinery →

Cleaning control →

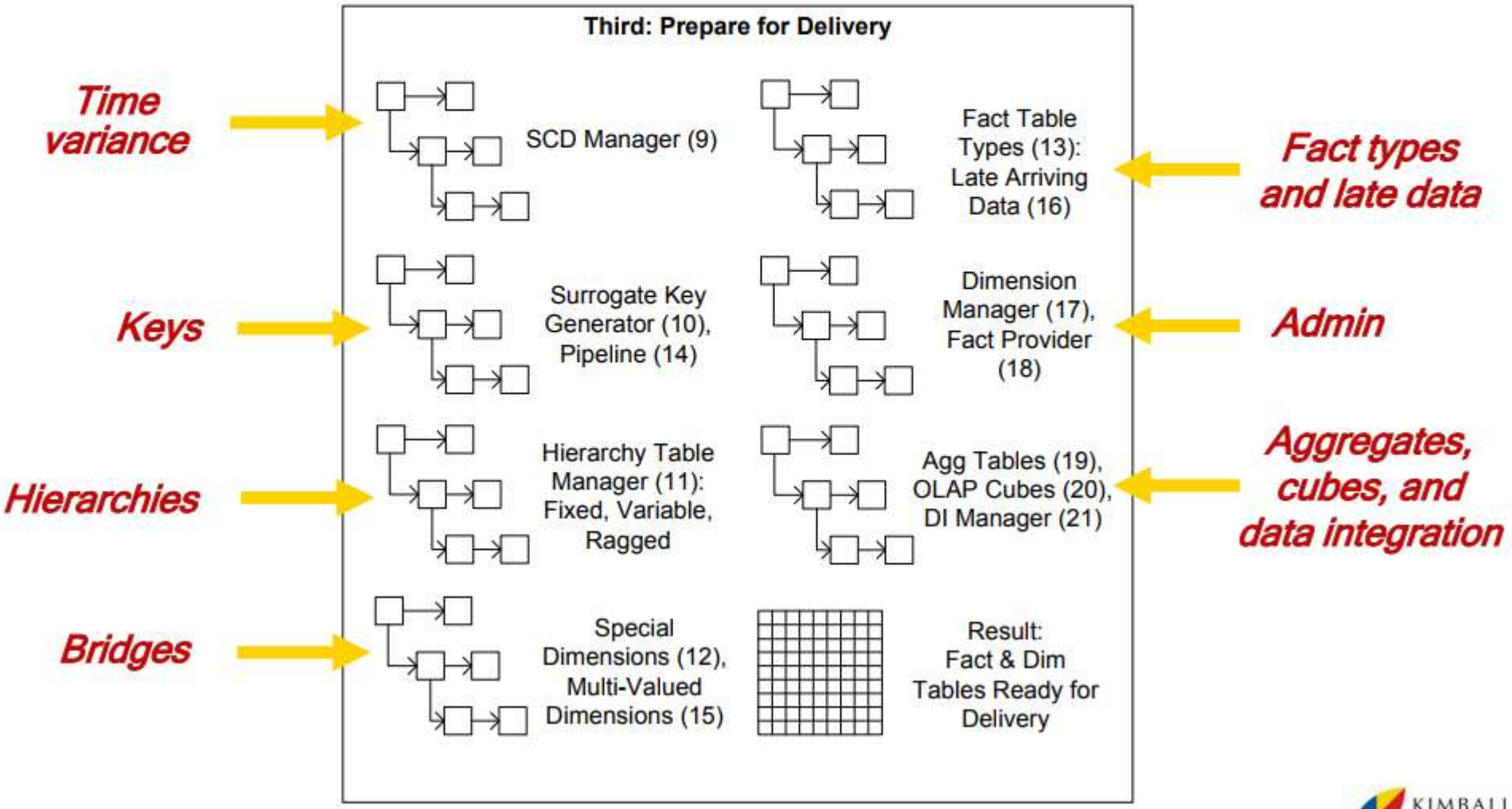
Integration →



KIMBALL GROUP

L: Prepare for Presentation

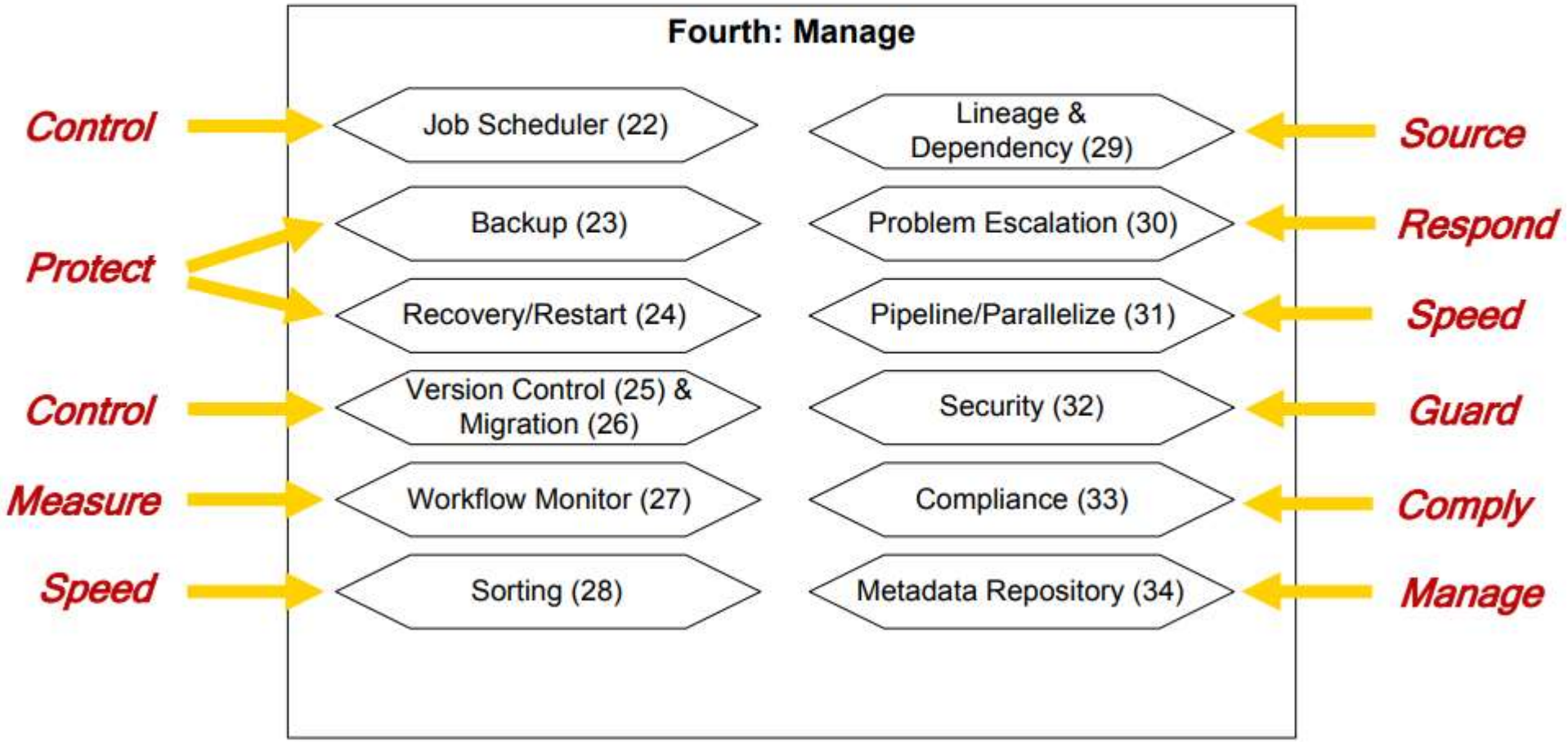
Note: Numbers in the parentheses refer to Kimball's 34 ETL subsystems.



KIMBALL GROUP

M: Manage All the Processes

Note: Numbers in the parentheses refer to Kimball's 34 ETL subsystems.



KIMBALL GROUP

Extract Subsystem

- Data Profiling – talked about that a couple weeks ago. We'll talk about it again tonight.

- Change Data Capture - This is a fancy way of talking about updating the data warehouse. In some data warehouses there are elaborate methods to see what needs to be updated.
  - For ours, we don't need anything elaborate, because updates will be based on dates. Initial load will contain the whole year data; update will contain data after 12/31/17
  - The exception is if we notice an error in the data warehouse. We will want to fix the error in the source system and flag it somehow so that the next load replaces the old data in the warehouse with the updated data
  - There are also large data warehouses, where not all data can be refreshed at once. A system is needed to make sure the relevant data gets updated and that integrity errors are not introduced by only updating pieces

Extract Subsystem (2)/Transform Subsystem

- Extract – Move data from the source systems to staging database for further processing.

- Cleaning and Transformation – There's a lot to talk about. Here's a sample:
  - De-duping: Vendor lists for multiple stores may have names slightly different. When merging we need to eliminate duplicates
  - Same with customers (e.g. is A. Britzman the same customer as Tony Breitzman?)
  - Conforming data sources (e.g. Prices in Euros for the European operations and Prices in dollars for the US operations. Need to be standardized in one or the other or both)
  - Missing values – need to be filled in with default values, or someone needs to find the right values
  - Wrong values – 4 digit zipcodes need to be fixed, phone numbers in the wrong format etc.

Load and Management Subsystems

- We'll talk about many of these at a future date
- The SCD manager refers to slowly changing dimensions. For example if our Gallons of milk change to 120 ounces, that's a slowly changing dimension and there are strategies to deal with those. (But we don't need to talk about them tonight)

- Surrogate key generator – we will talk about this. We need to generate keys to map our dimension tables to fact table. They need to be random/meaningless and should be integers. It's the ETL systems job to generate them

- OLAP/Cube Aggregator – that's a subject for another night

- Metadata Repository – that's another subject for another night, but we will have to keep track of where our data comes from, so we will talk about it a little bit

The best way to talk about ETL…

- The best way to learn about ETL is to do ETL, so I have an elaborate team HW assignment to lay out.
- Basically I will put you in groups of 3 or 4 and each group will build a data-mart of 3-4 regional grocery stores, which will eventually get rolled up into an Enterprise Data Warehouse
- Since we have Data Analytics people, which may not be the best programmers, and CS people that are very good programmers, I have selected the teams so that some weak programmers are teamed with strong programmers
- Don’t worry, there will be enough non-programming to do so the programmers will not do all the work

Teams

Name	Store Key
Alacqua, Joseph B.	105
Seedorf, Robert J.	111
Adieze, Chibuike S.	113
Kamal, Takeshwari	211
Das, Plaban	234
Kalatsjov, Jevgeni	267
Sridharan, Harini	289
Robertson, Cole T.	301
Black, Clifford	313
Carlin, David J.	327
Smith, Abigail M.	388
Saunders, Johnathan A.	405
Rivera-Lau, Stephen J.	419
Murshed, Ridwan	423
Vivona, Michael	444
Smith, Ryan W.	509
Leonchuck, Michael B.	515
Madala, Himabindhu	534
Schneider, John A.	578

- Take 2 minutes and introduce yourself to your teammates
- If you can't find your team, I will help
- Note Ridwan is a PhD student with a conflict. He doesn't come to class, but is very good. You will be glad he is on your team and he will make a contribution
- Mohammad Motemedi is just auditing the class. Mohammad, you can hang out with whatever team you like and can contribute as much or as little as you like

Key Steps

- Step 1 - Data Profiling: You will want to do a series of profiling steps to make sure your store databases are reasonable
  - For example if Froot Loops are the 10th best selling item in Bob's Market, but they're the 900th ranked item in Joe's Market then there is something wrong with one of the programs.
  - (By the way, each store should have a name, and the individual team member who generated the data will be listed as the store manager. You can make up an address and other info as shown in the table several slides ahead.)
  - Also, profile sales, customers per day, total transactions and make sure differences among stores make sense

Key Steps (2)

- Step 2 – Fix individual programs
  - If there are problems with one or more programs, then team members should help other team members fix any bugs
  - Do not just replace all programs with the best working one. We want different programs to have different output formats etc. So fix the existing programs unless they're completely hopeless.
    - The idea here is that chains of stores will have scanners from different manufacturers or different eras and will consequently have different feed formats. ETL teams have to deal with data in multiple formats
  - Next re-run profiling exercise.
    - Check for missing values, missing dates, etc.
    - Recheck top selling items as before
    - Deliverable 1 should look something like the next slide

Deliverable One

Combines Sales (December)			Joe's Market		Sue's Market		Jane's Market		Steve's Market	
$ 13,043,675.00			$ 3,069,100.00		$ 3,682,920.00		$ 3,376,010.00		$ 2,915,645.00	
Combined Customer Count (December)			20400		24480		22440		19380	
Best Selling Items Combined		# Sold	# Sold	Rank	# Sold	Rank	# Sold	Rank	# Sold	Rank
SKU 1	Product 1	130050	30600	1	36720	3	33660	2	29070	1
SKU 2	Product 2	127415	29980	2	35976	2	32978	1	28481	3
.										
.										
.										
SKU 25	Product 25	63707.5	14990	25	17988	24	16489	28	14240.5	23

A Common ETL Issue

- Missing Data/Conforming Data – The products table has multiple issues
  - Some ItemTypes are missing
  - Some ItemTypes are too broad
  - Some ItemTypes are too fine

- Upper Management has decided that every SKU should be mapped to one of the 114 product subcategories in product_class.txt

- This happens all the time. Companies get reorganized into new business units and the Data Warehouse has to get reorganized as well
  - It's a pain, but remember a Data Warehouse is for business users in management. They don't care about inconveniencing the ETL team

Fixing the Products Table

- The Good News: 1319 of the 2076 products have an itemType that maps directly to the subcategory

- The Bad News: 294 items in the product table have null values. These will have to be assigned to one of the new subcategories
  - is there a way to automate this, by string searching or is it a data entry problem? ETL team decides.
  - Are there any products that don't fit into a category? Suggest a new category to management (me) if you find one that doesn't fit.

- More bad news: There are 597 items with item types, but those item types are obsolete (either too broad or too fine).
  - For example There are Numerous Thomas's English Muffins that are categorized as 'Baked Goods Other Than Bread'. These want to go to the 'Muffin' subcategory
  - Similarly there are a bunch of Potato Chips currently called 'Snacks' that want to go to 'Chips' and a bunch of 'Bread' that want to go to 'Sliced Bread'
  - I'll bet there is a way to automate these changes

Deliverable 2a

- Deliverable 2a: A new products table that contains: Manufacturer, Product Name, SKU, Size, Product Class ID, etc as shown
  - This should have no null values (and only Product Class IDs that exist in the Product Class Table or new ones that Management has agreed to add)

Product Dimension	
Product key	int
SKU	ShortText
Product Name	ShortText
Product Class ID	int
Subcategory	ShortText
Category	ShortText
Department	ShortText
Product Family	ShortText
Size	ShortText
#Per Case	int
Brand Name	ShortText
Manufacturer	ShortText
Supplier	ShortText

- Note supplier is always Rowan Warehouse for everything but milk. Milk comes from Rowan Dairy
- Note the product key is a random integer for each SKU. (You might want to ask other teams how they are generating theirs, otherwise when we roll up all of the data-marts into an Enterprise Data Warehouse, we will have an issue)

Deliverable 2b

- Deliverable 2b: We haven't talked about meta-data much, but every record needs a source and a reason.
  - So we'll need another table that contains each SKU and source#
  - Source# can just be 1,2,3,4,5, etc. where 1 means it came from original product table, 2 means it was mapped by hand by Jane Doe, 3 means it was done by a string match like Product Name = 'Frito Lay' implies subcategory 'Chip' etc.
  - You will obviously need a source# table that contains definitions similar to above for each source#

Another Common ETL Problem

- It's likely that if you have 4 stores that you may have more than one date format (e.g. 20170101, 1/1/2017, January 1, 2017
- Make sure as part of your ETL process you conform the dates.
- Don't rewrite the individual programs to use the same format (that's cheating). Instead conform the dates in the staging database that combines the store data
- You might ask the other teams what kind of date format they're using so we don't have to do this again when we roll-up the data-marts into an Enterprise Data Warehouse (Maybe we should vote)
- Note this is why Bill Inmon (father of data warehousing) doesn't like building Data Warehouses bottom-up. He believes the Data Warehouse should be built first, and then individual data marts should be subsets

Deliverable 3

- This one's an easy one

Store Dimension	
Store key	int
Store Manager	ShortText
StoreStreetAddr	ShortText
StoreTown	ShortText
StoreZipCode	ShortText
StorePhone#	ShortText
StoreState	ShortText

- Store key is on an earlier slide
- Manager name is you
- Other fields can be made up

Deliverable 4

- Another easy one

Date Dimension	
DateKey	Int
Date	Date/Time
DayNumberInMonth	int
DayNumberInYear	int
WeekNumberInYea	int
MonthNum	int
MonthTxt	ShortText
Quarter	int
Year	int
Fiscal Year	int
isHoliday	Boolean
isWeekend	Boolean
Season	ShortText

- I would make the datekey 1 to 365. (a 2-byte int should allow for 50 years so that ought to be enough)
- Someone will have to look up the holidays and fill in the appropriate fields
- Base seasons on the Solstices and Equinoxes
- Fiscal Year ends in July so anything before July 31 is 2016 and anything after is 2017

Deliverable 5

- Not Really a deliverable (don't send it to me)

Sales Fact (Transaction Level)	
CompositeKey	ShortText
DateKey	Int
ProductKey	Int
StoreKey	Int
SalePrice	float
CostToStore	float
GrossProfit	float
Transaction#	int

- Just do it for the month of December
- Make transaction # an 8 digit number (leftmost digit is a 1; next 4 digits are customer # for the day, next 3 digits are transaction #. So if the item is the 5th item bought by the 7th customer of the day, the transaction # would be 10007005)
- Composite key is 8 characters consisting of 3 digit store key, 3 digit (001 to 365) padded date key, 4 digit product key (in that order)
- Cost to store is base price; profit is sale price minus base price
- Note the composite key is unique even though product key, transaction key, etc. are not

Deliverable 6

- This one is a daily aggregate of the individual transaction table

Sales Fact (Daily Level)	
CompositeKey	ShortText
DateKey	Int
ProductKey	Int
StoreKey	Int
#SoldToday	int
CostOfItemsSold	float
SalesTotal	float
GrossProfit	float

- Note you can't use your Deliverable 5 for this though because that only has a month's data
- A better approach would be to aggregate from the individual stores and then append them together

Deliverable 7

- I don’t think I want the whole table, but build the whole table
- We’ll decide on the deliverable at a later date

InventoryFact (Daily Level)	
CompositeKey	ShortText
DateKey	int
ProductKey	int
StoreKey	int
#Available	int
CostToStore (itemLevel)	float
CostToStore (caseLevel)	float
#CasesPurchasedToDate	int

Deliverables 8-11

- 4 Quarterly inventory snapshots

InventoryFact (Quarterly Snapshot)	
CompositeKey	ShortText
ProductKey	int
StoreKey	int
Quarter and Year	ShortText
Quarter	int
Year	int
#CasesPurchasedToDate	int
#CasesPurchasedThisQuarter	int
#CasesOnHand	int
TotalCostToStoreThisQuarter	float
TotalSoldByStoreThisQuarter	float
TotalCostToStoreYTD	float
TotalSoldByStoreYTD	float

Due Dates

- Due Dates:
  - Deliverables 1-4 November 3
  - Others TBD

Tips/Tricks (1)

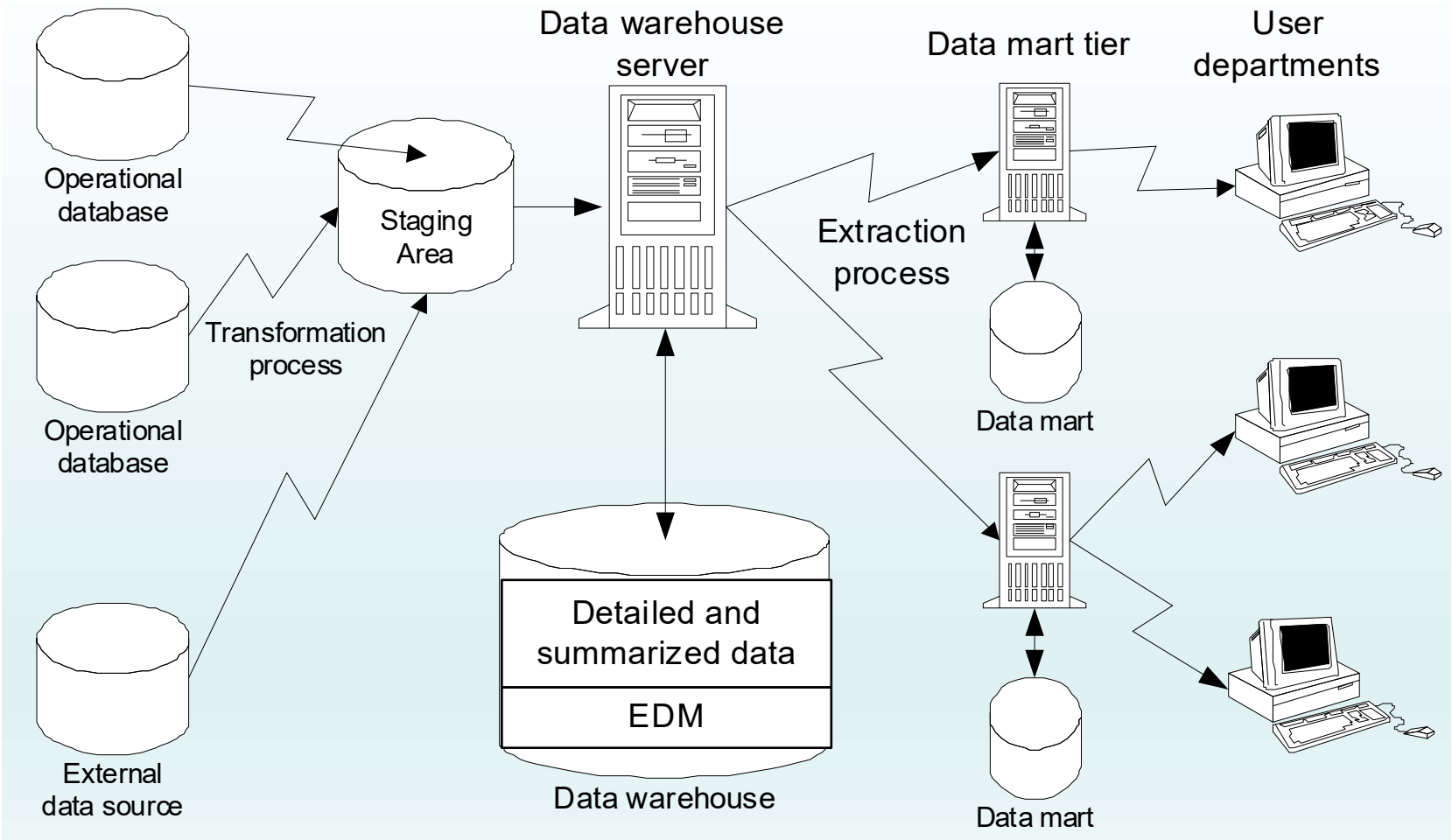
- Document as you go
- We will eventually put documentation into the Metadata repository
- Note we built dimension tables first.
  - This is not an accident.
  - Keys are made up and generated at the time the dimension table is created
  - Fact tables use these keys. Make sure you have referential integrity (that is there is no dimension key in a fact table that is not in the appropriate dimension table and vice-versa)
  - While we're at it, check every field of every table and make sure there are no nulls
- There is no such thing as an ad-hoc query in a data warehouse
  - It may sound ridiculous, but every time you think of building an ad-hoc query don't do it. Every query should be called from a stored procedure or macro or trigger or batch file, or from a glue language like PowerShell or Python or from an integration tool like Pentaho Data Integration

Tips/Tricks (2)

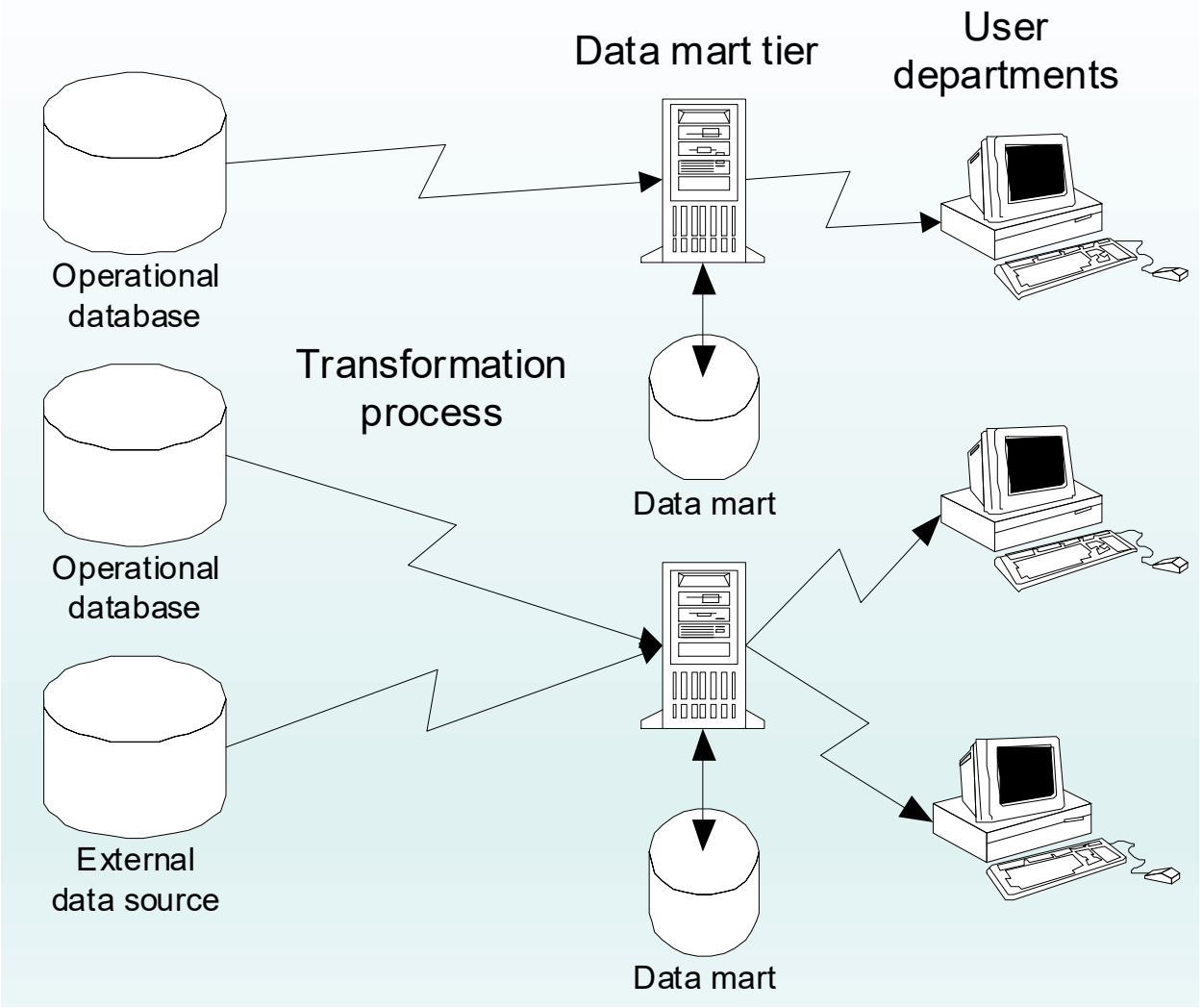
- Keep track of sources of tables
  - You don't have to go to the detail of deliverable 2b (that's an exercise to show how the world works)
  - But we should know the source of the transactions (e.g. JoeGroceryV3-6.py) Because although it's fresh in your mind now, which version of the program is the latest and greatest, that might not be the case a month from now.

- Each team is responsible for its regional datamart of 3 to 4 stores, but it might be worth talking to the other teams about their approach as you progress. Remember we are eventually rolling these things up, so the better they are integrated now, the easier it will be to build the final warehouse

This is Not what we're doing
Top Down Architecture (U. Colorado Bus. School)



This is what we're doing
Bottom Up Architecture (U. Colorado Bus. School)



- No centralized warehouse. Inmon argues this is a bad idea. Kimball argues this is a good intermediate step to building a warehouse

One More Thing… (maybe more than one)

- I know that many of you hate group work...
- Too bad!

- There is no such thing as an ETL Lone Wolf. It's called an ETL team. You may find yourself on one some day.

- If you are feeling overwhelmed at this point, relax. It's a lot of steps, but I deliberately put it into bite size pieces. None of the steps are particularly difficult (tedious perhaps but not difficult).

- The difficult part is done

- I'll bet you never had a HW assignment that took an hour to describe before!