Teradata Physical Design Process

Agenda

- What is a Physical Design
- Why is a Process Necessary
- The LDM to Physical Design Process
    1. Preparation
    2. Remove Unmapped Entities (Tables)
    3. Remove Unmapped Attributes (Columns)
    4. Consider Key Only Entities
    5. Assign Data Types
    6. Consider Collapsing Sub Types
    7. Add Processing Columns
    8. Current & History Tables Creation
    9. Local Alignment
    10. Primary Index Validation
    11. Secondary Index Consideration
    12. Other PDM Activities
    13. Walkthrough

What is a Physical Design (PDM)?

- A specific Client Implementation of a Logical Data Model
  - It must end up **looking similar**, not identical to the customer LDM or you have wasted your effort
- Includes **physical** attributes and needs like:
  - Audit capability
  - Full time variant update
  - Performance and availability
  - Security
- It caters for real world limitations like:
  - The fact dates from source are very often of poor quality
  - Referential integrity for master data only exists in the minds of data modelers
- Concepts like Tables, Columns and Indexes
  - Should have fully defined properties (ie columns with data types and domains and defaults for processing considerations)
  - Primary Index design for performance
- The PDM is the physical implementation of the business rules and requirements embodied in the LDM
- The PDM is the physical implementation of the logical data structures needed to support the analytic needs of the business

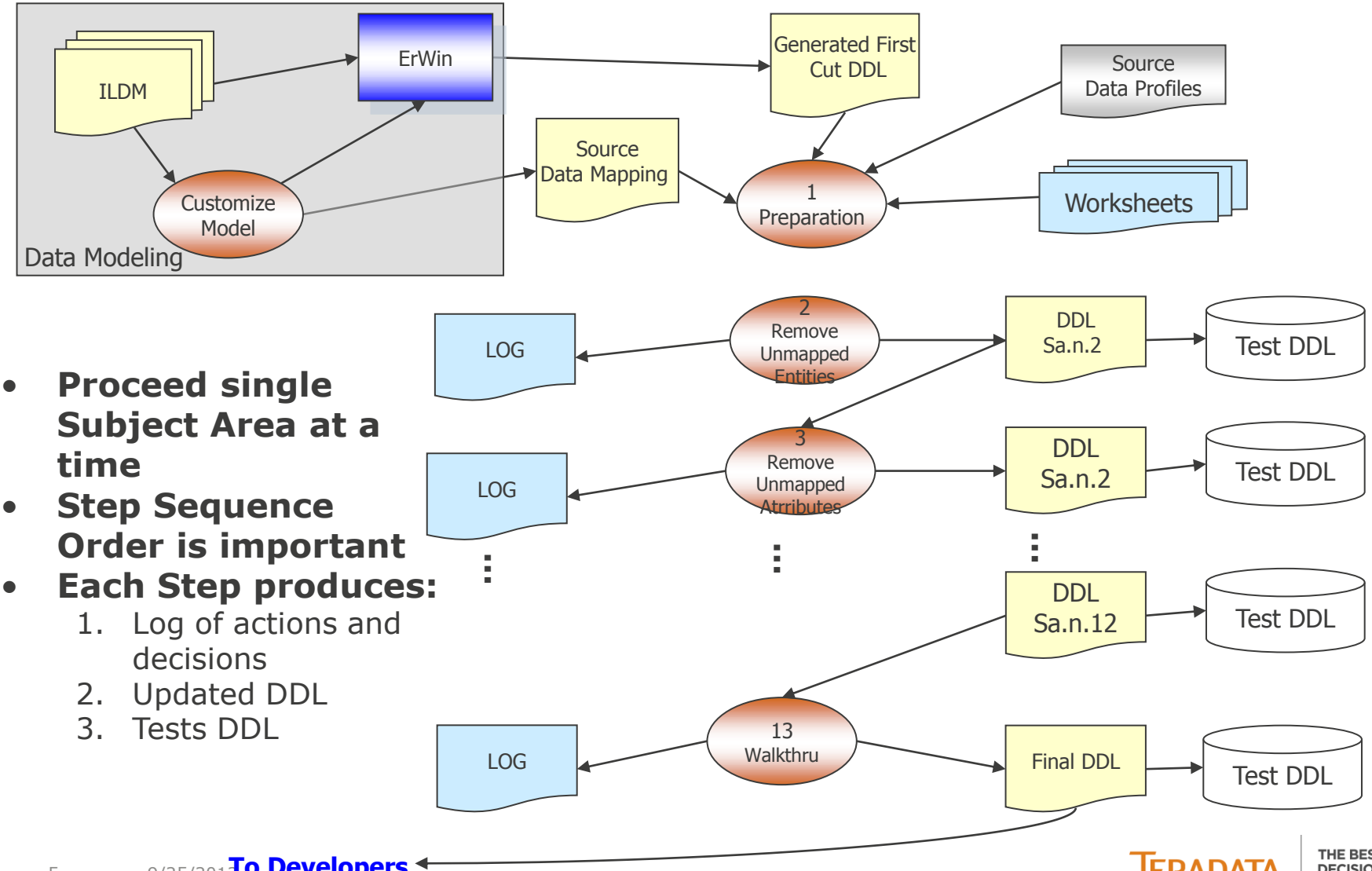
Why is a Process necessary?

- So the database design is based on sound requirements and architectural principles
- So the PDM is based on the customer and database reality (facts) not generic theory
- To give consistency & repeatability
  - Reduces implementation project risk
  - Reduces ongoing support
  - Enhances scalability
  - Minimises impact of changes in source systems
- To allow for Localization
- Because performance tuning considerations do not exist in an iLDM or a customer LDM

Why is a Process necessary?

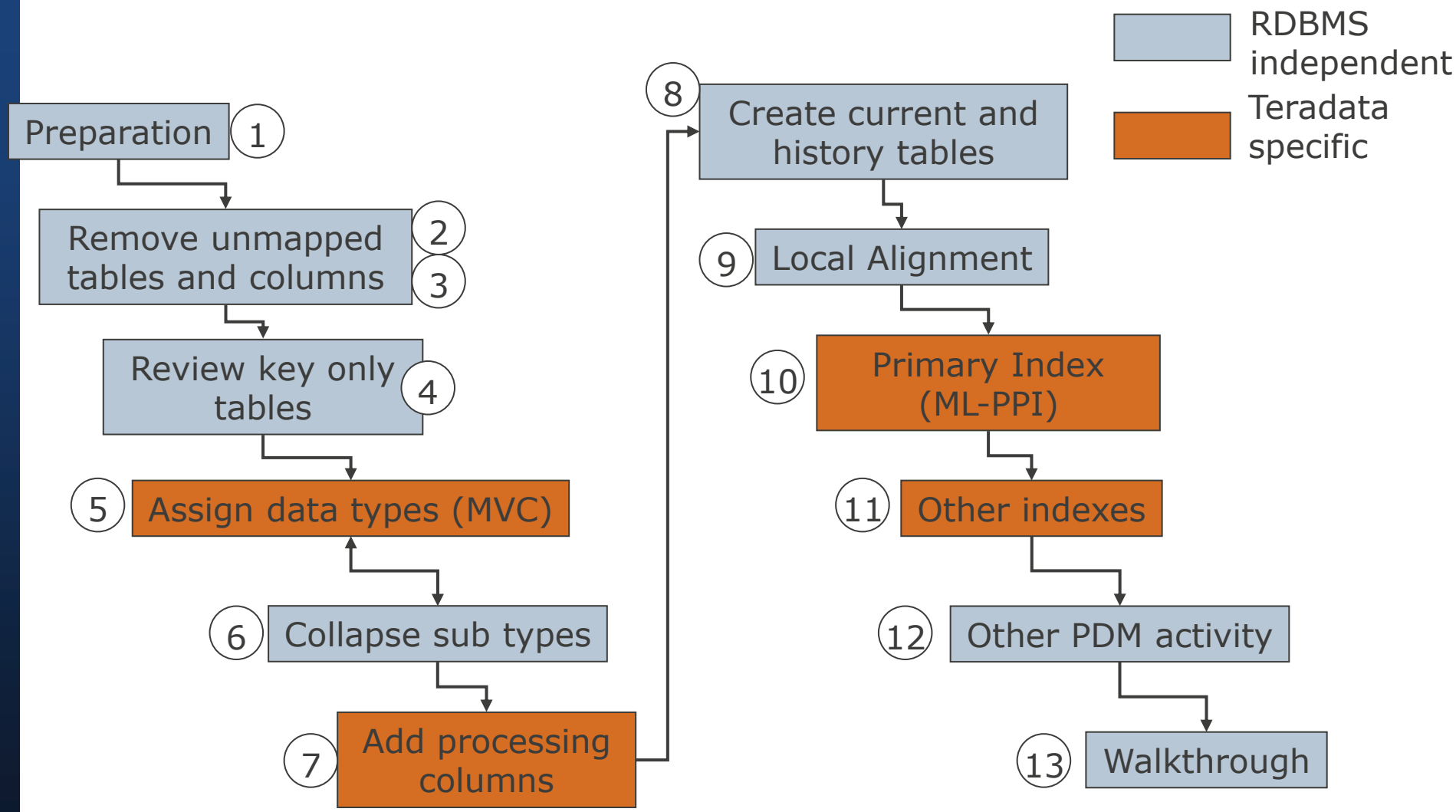
- Provides a standard approach to PDM creation
- Breaks a big job into smaller tasks - a Subject Area at a time
- Adjusts tool (i.e. ERWIN) generated DDL from the customer LDM, a good starting point but needs to be improved due to:
  - Primary Keys not Primary Indexes
  - Potentially Wrong Data Definitions
  - Number of Never Used Tables
  - Number of Never Used Columns
  - Non-Aligned Column names
  - Key Only Tables of small value
  - No Audit or Processing support
  - Difficult Model for end users to traverse due to extreme normalization
  - Potentially No Naming Standards
  - Not fully defined Domains

The Process Overview and Positioning



- **Proceed single Subject Area at a time**
- **Step Sequence Order is important**
- **Each Step produces:**
  1. Log of actions and decisions
  2. Updated DDL
  3. Tests DDL

Waterfall process per subject area



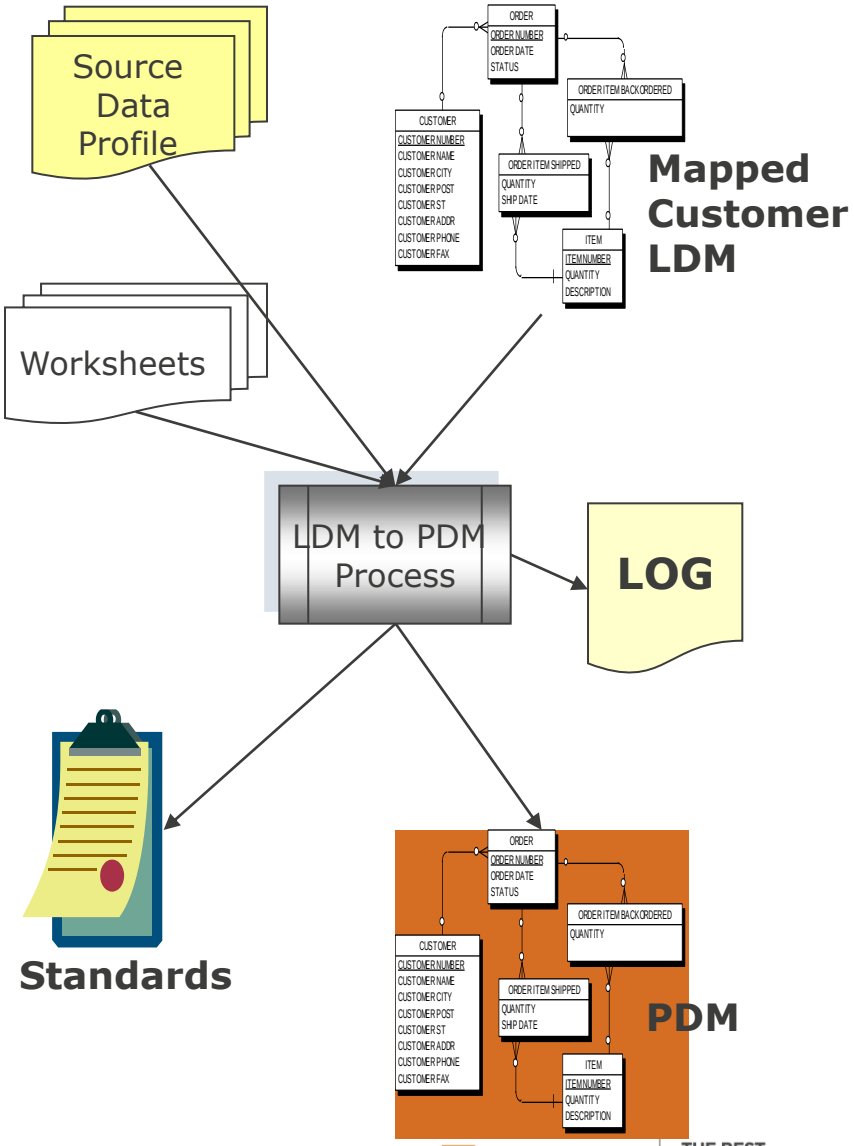
I will not be presenting rules…

It's too easy to get "preciousssss" about topics like Database Design

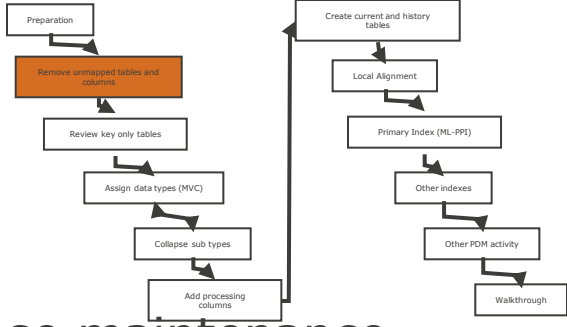
- Each implementation IS different
- All or NONE of these may apply in any design or implementation
- Treat the messages in this presentation as decision points – make auditable decisions!

1. Preparation Process inputs and outputs

- Inputs
  - > Source data profiles
    - – Data quality
    - – Data availability
  - > Worksheets
    - – Mapping
    - – Transforms
  - > Mapped customer LDM
  - > ETL SLA
  - > Business Questions
- Outputs
  - > The process Log
    - – Enables process restart
    - – Logs all changes from customer LDM
  - > Standards
    - – Defaults
    - – Domains
    - – RI
  - > The PDM itself
    - – Tables
    - – PI's, PPI, MVC
    - – JI's, SI's

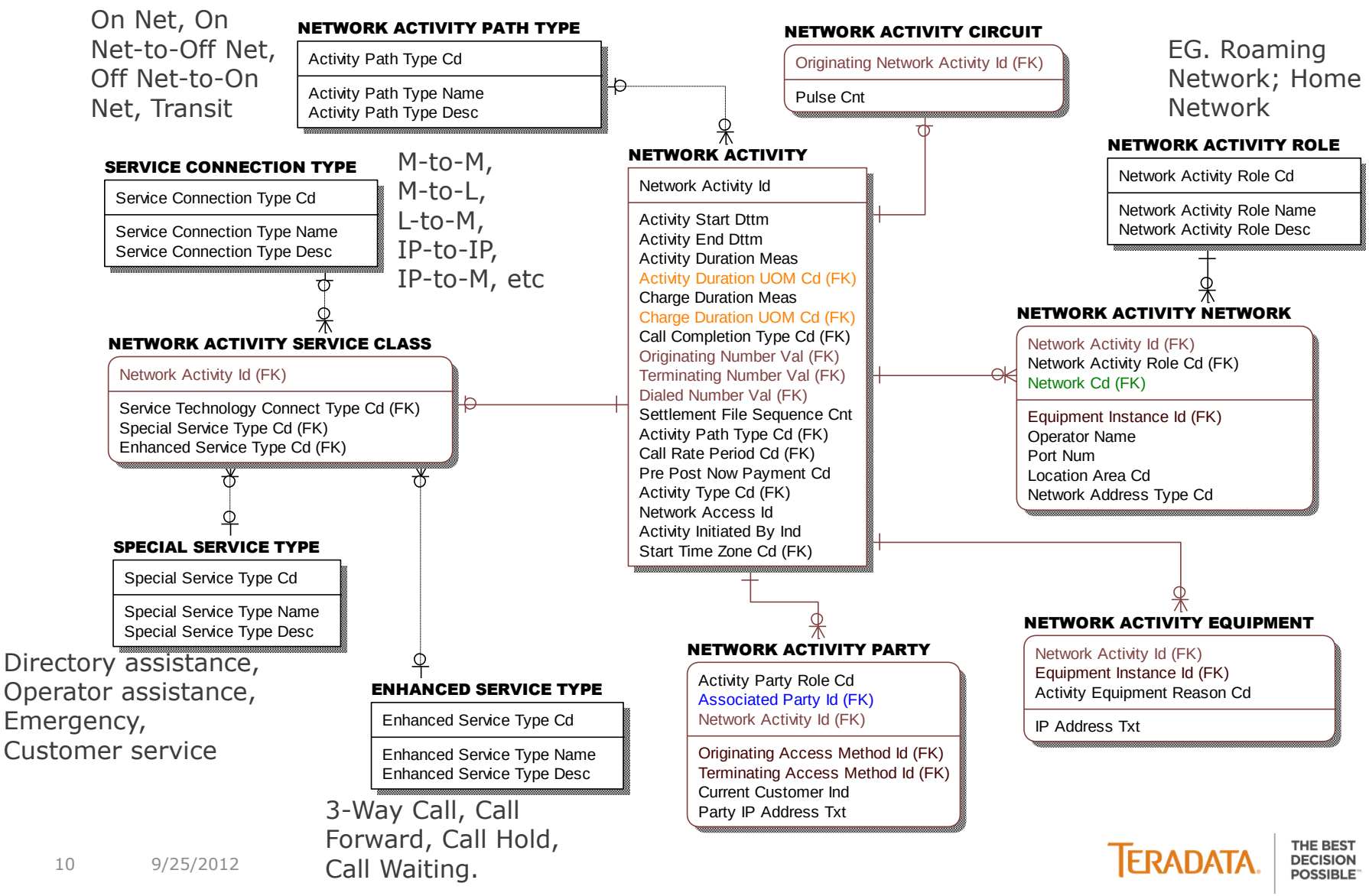


2. Remove Unmapped Tables

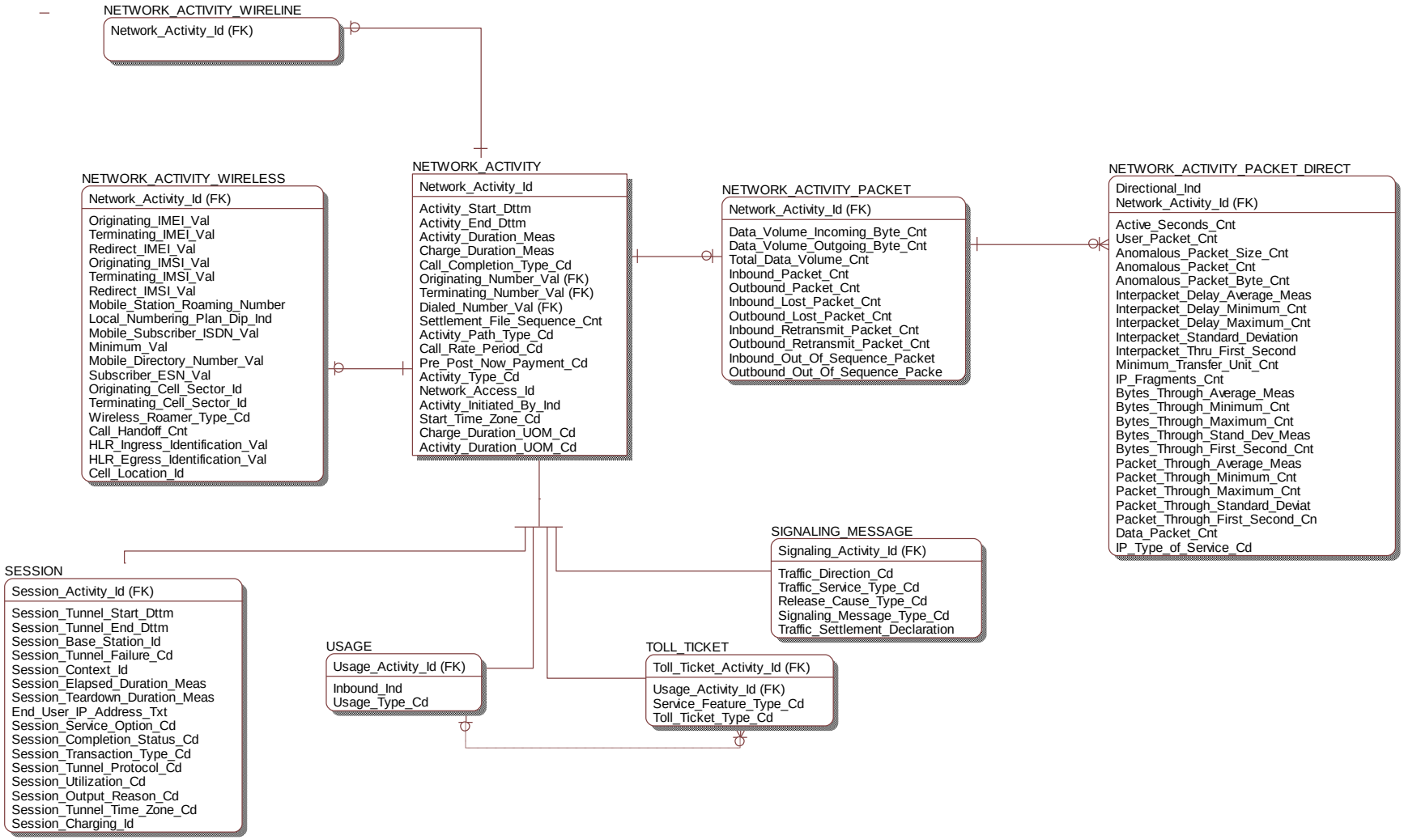


- Why?
  - > Unused tables and columns confuse and increase maintenance effort
  - > Serves to verify the mapping
- Process
  - > Mark every column not mapped as Logical Only in every table
  - > List all tables with no columns mapped
    - – Exclude from consideration 'Key Only Entities' that link tables
    - – Complete a 'Null Entity' Worksheet for each such entity
  - > Verify that there is no data for these tables with the Client and with the modelers (as it could be an error)
  - > Remove the tables from the PDM
    - – Do not remove any entities or attributes from the LDM
  - > Add an entry in the Transformation Rules Worksheet for **ea** entity identified as empty

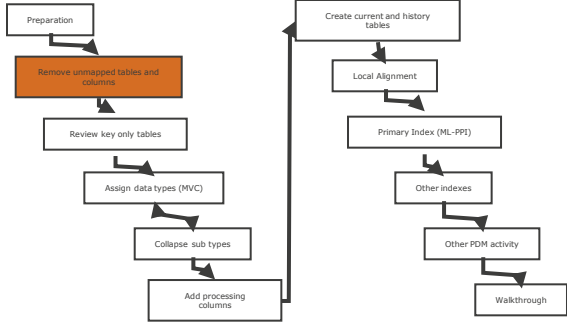
Network Activity Base



Network Activity: Mapped Entities

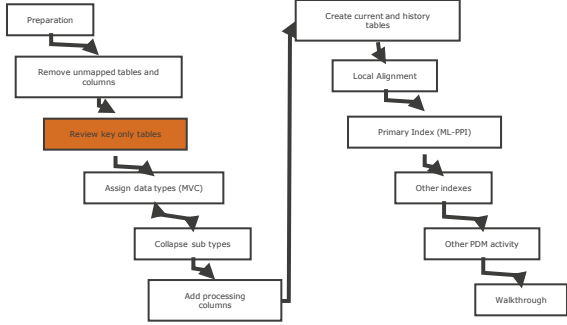


3. Remove Unmapped columns



- Why?
  - > Un-mapped columns in the PDM confuse and increase maintenance effort
  - > Serves to verify the mapping
- Process
  - > Mark every column in remaining tables that are not mapped
  - > Complete the 'Not Mapped Attribute' worksheet
    - – Review with the Client all such attributes
  - > Add an entry in the 'Transformation Rules Worksheet' for **each** attribute identified as not mapped
  - > Remove the columns from the PDM
  - > Does it look like it should be mapped?
    - – Key component not mapped? Possibly composite key part?
    - – Is it a Foreign Key?
    - – Code value? Not identified during mapping?
    - – Obvious values not mapped and need checking. E.g. Birth_Dt
    - – Ask the Data Modeler to verify

4. Consider Key Only Entities



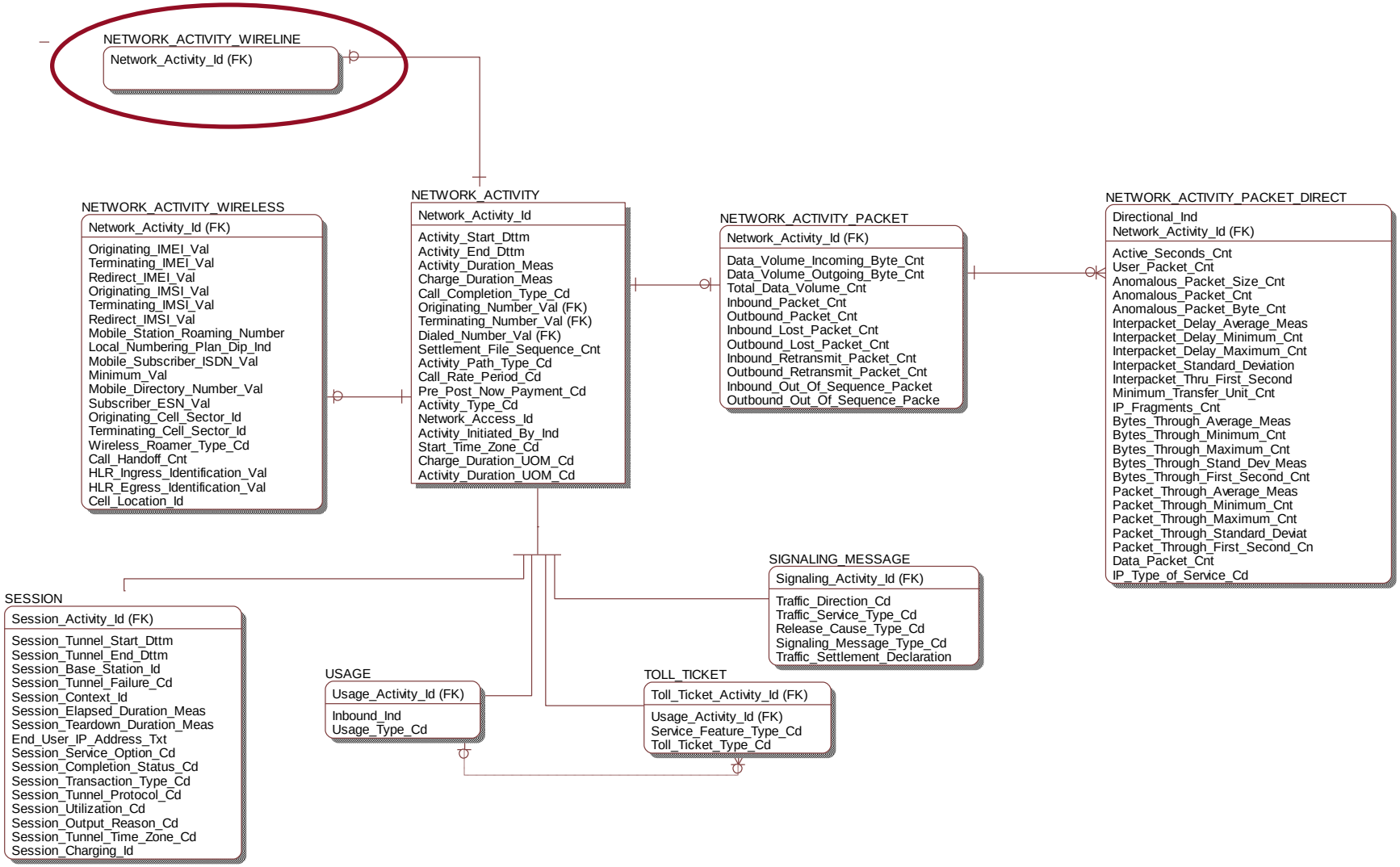
- Why?
  - > Key Only Entities may be included for navigation
  - > Alternatives are available when designing the PDM to overcome the need for navigation Key Only Entities (e.g. view or join index)
  - > A Key Only Entity may be easily stored as a value in the original table

- Process
  - > Highlight all key only entities
    - – Do they add any value?
    - – Can they be collapsed into the original table?
    - – Identify the business rules and requirements that depend on this entity. If you can't identify the requirement then the entity may not be needed.
    - – Is denormalisation an option?

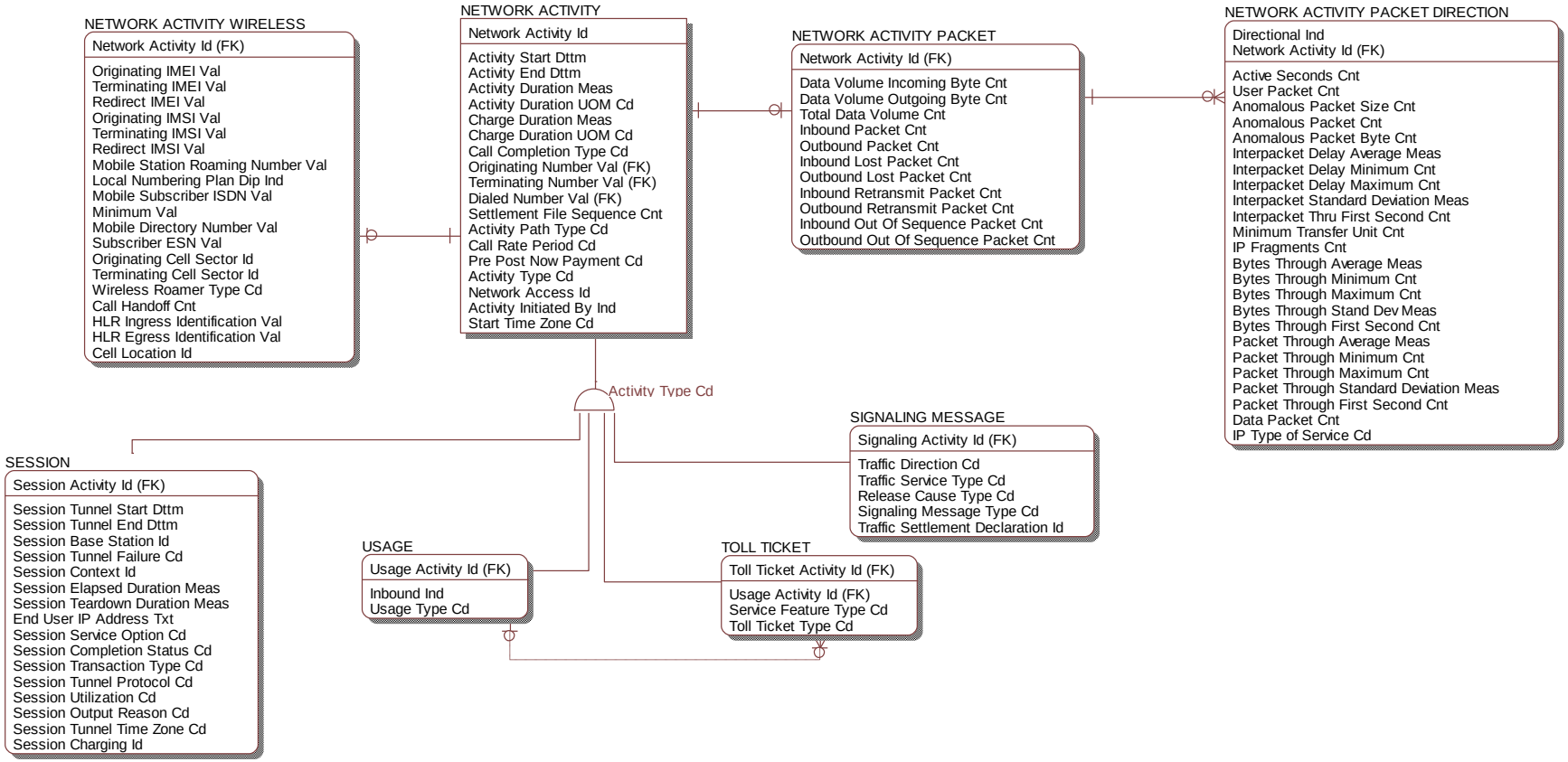
  - > Update the Transformation Rules worksheet with the **reason** for every change.

  - > Update the PDM

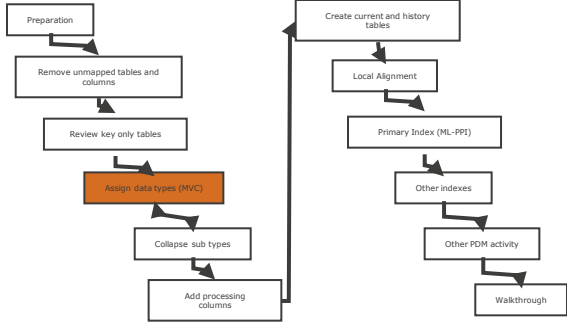
Network Activity - Key Only Entities



Network Activity Logical



5. Assign Data Types (Domain Enforcement)

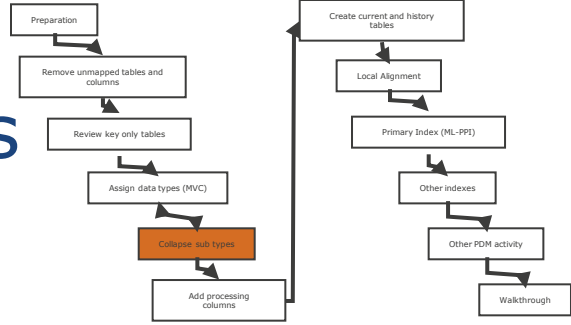


- WHY?
  - > Most customer LDM's do not have fully defined and consistently applied domains
  - > Poorly defined column data types lead to poor join performance, unpredictable calculation, sort and selection results
  - > Inconsistent definitions create programming and testing overheads
  - > Performance and inconsistency issues will create testing delays and conflicts
  - > Users may loose confidence in the EDW (a prime cause of failure)
  - > Try to avoid NULLs in columns
    - – Users do not understand what NULL means
    - – Joins on NULLs give poor performance
    - – NULLs can lead to major skews in temporary or intermediate tables

5. Assign Data Types (Domain Enforcement) - continued

- Most important for PIs or other join-columns in large volume tables:
  - Agreement
  - Event
  - Account_Balance_Summary_DD
  - Party_ID
- Most efficient joining is on INTEGER
- Least efficient is on VARCHAR – change to CHAR and compress
- Joining different data types is SIGNIFICANT overhead (see Explain on previous page)

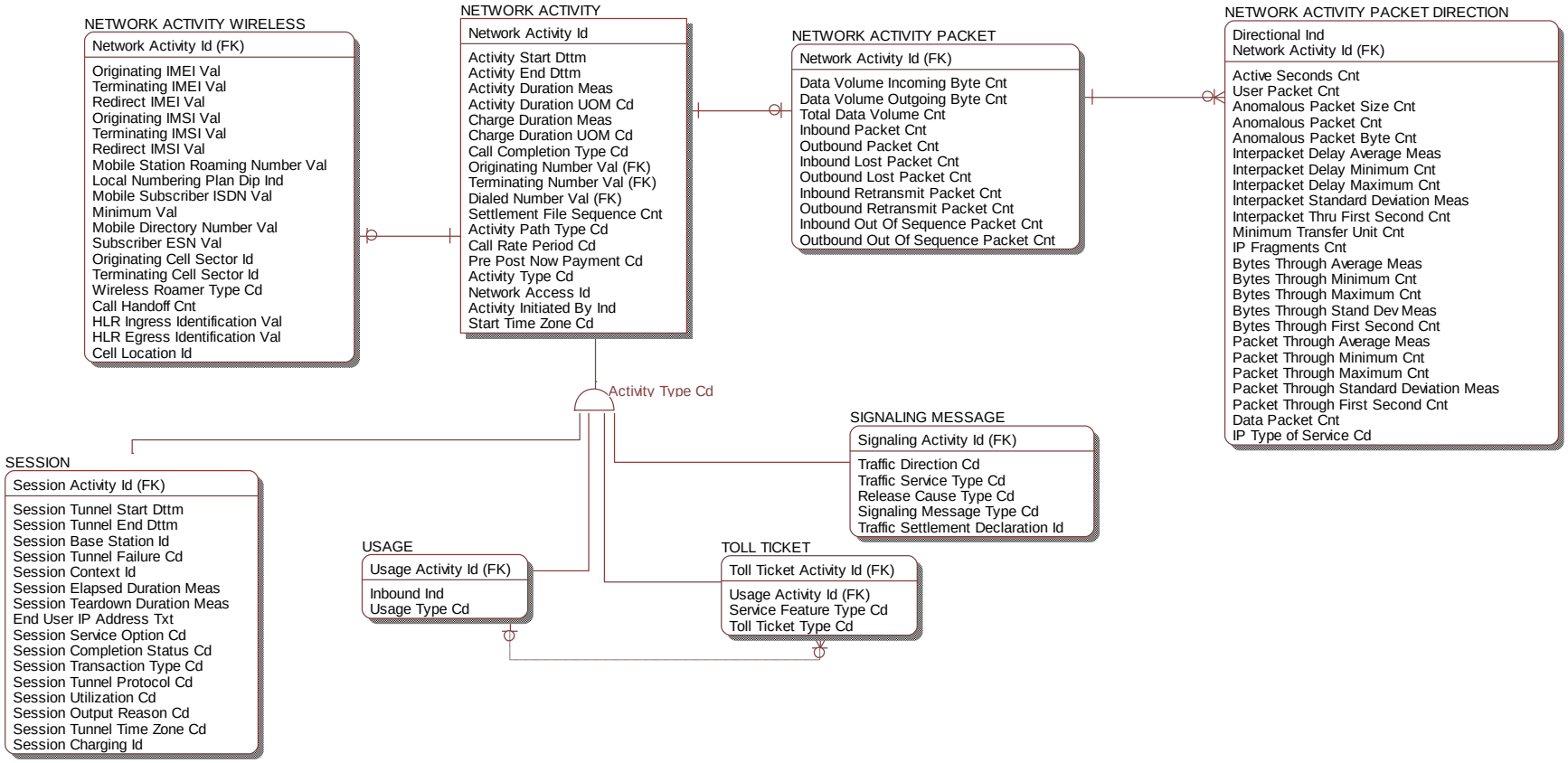
6. Consider Collapsing Sub Types



- Why
  - > Few data columns per subtype
  - > Subtype data columns generally required for many queries
  - > If from the same source, it is often simpler to load once
  - > MVC removes one of the main physical reasons for establishing subtypes: space taken by unused columns

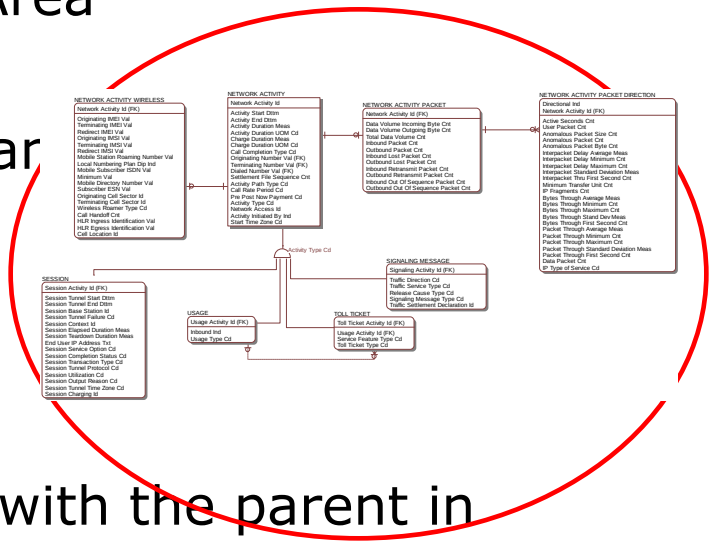
- Process
  - > Review subtypes in sets
    - – If many columns unlikely candidate as all non common columns must be moved up to the parent
  - > Analyse data that is required **often** for queries, these attributes may be candidates for moving back to parent, even if not full de-normalisation.
    - – What is the demographic across sub types, 90% in one type?
  - > Change the model according to results.
  - > Entry in the Transformation Rules worksheet for each alteration taken with full justification.

Network Activity Logical



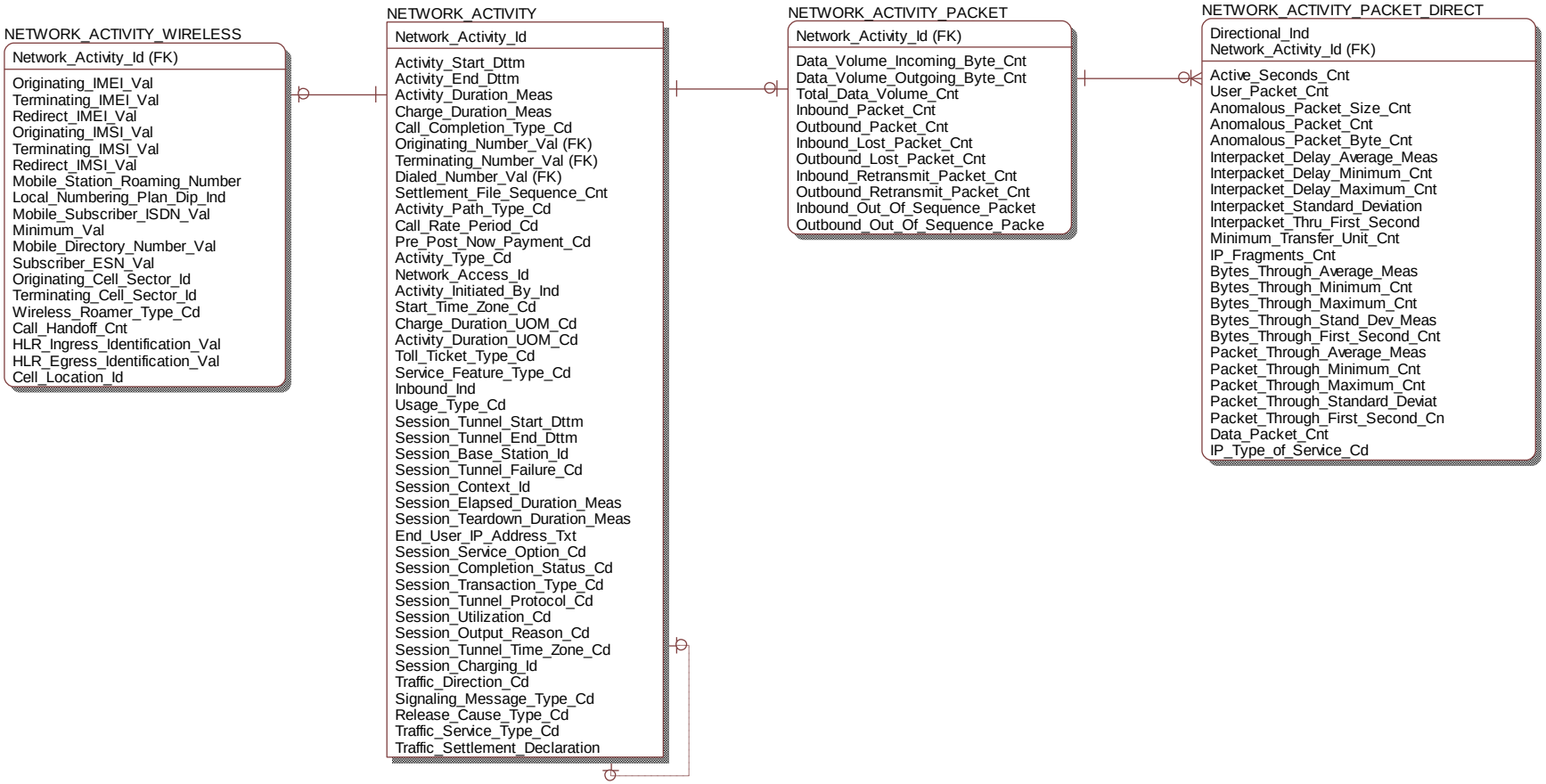
6. Consider Collapsing Sub Types – Workshop

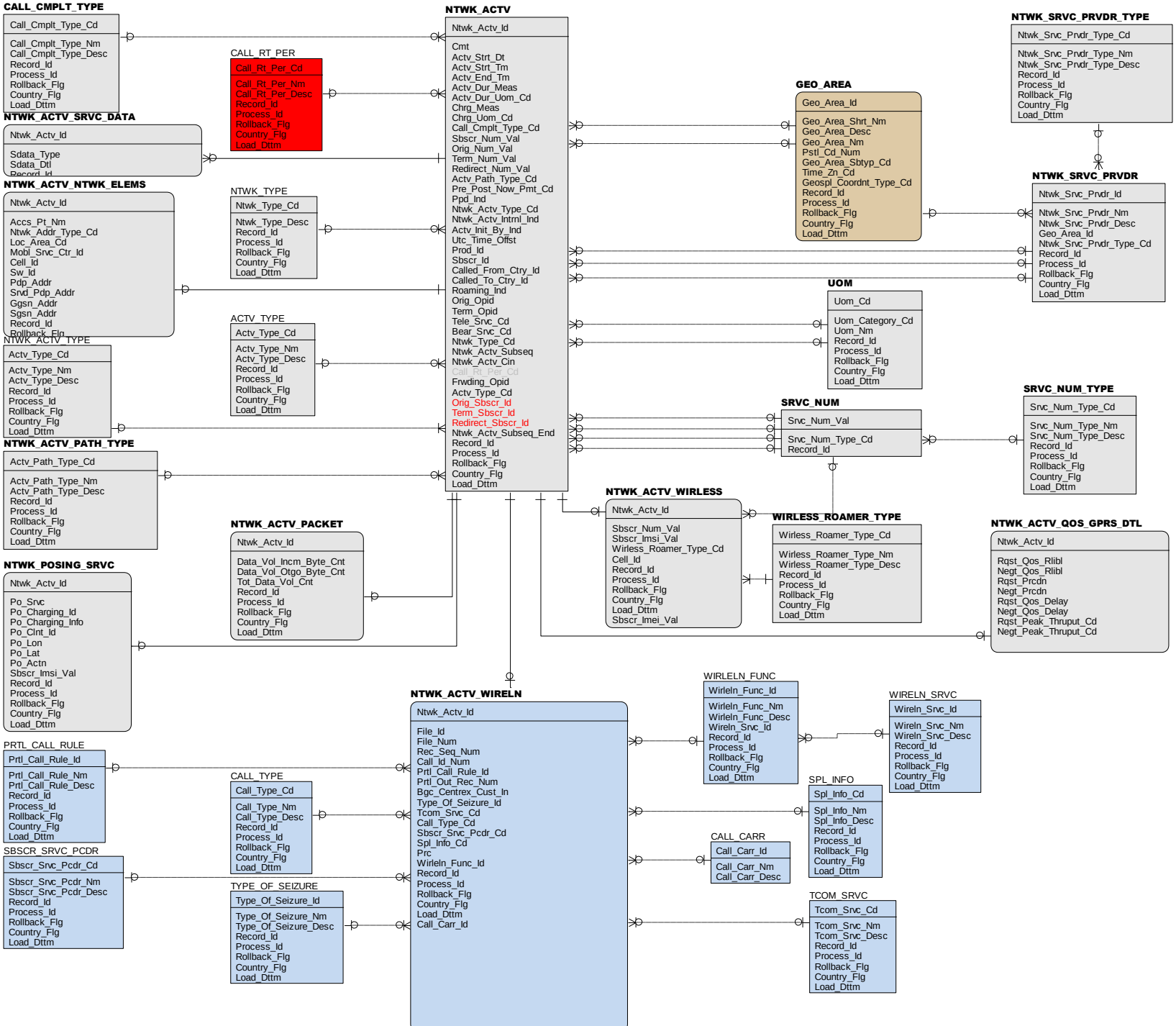
- E.g. cLDM Network Activity Subject Area



- How often do the attribute values char
  - > Never
- Is your data warehouse time variant?
  - > Yes
- Do you mostly use the subtype data with the parent in queries?
  - > Yes
- Consider collapse sub-types into Individual parent table..

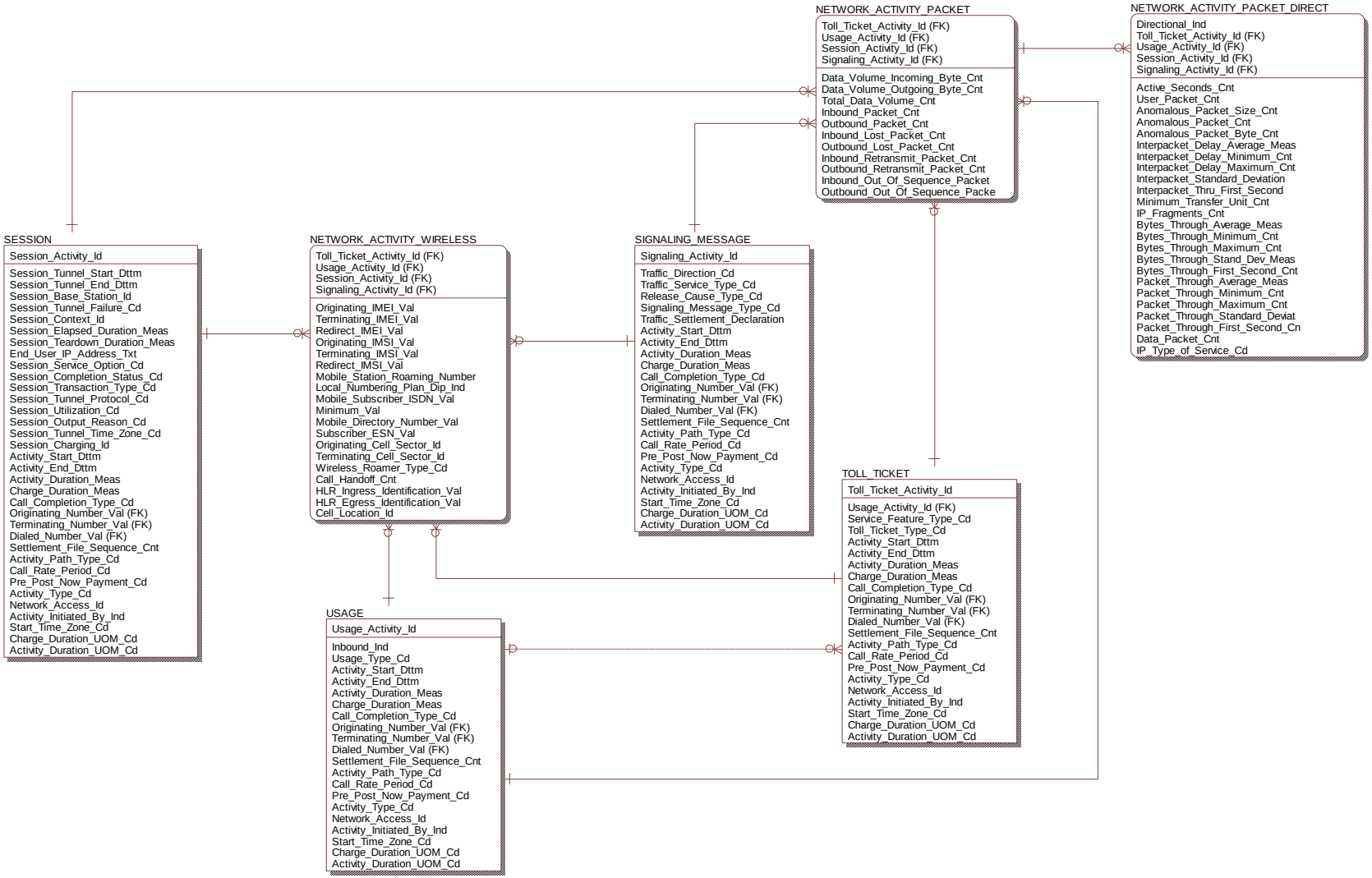
Network Activity Sub type roll up

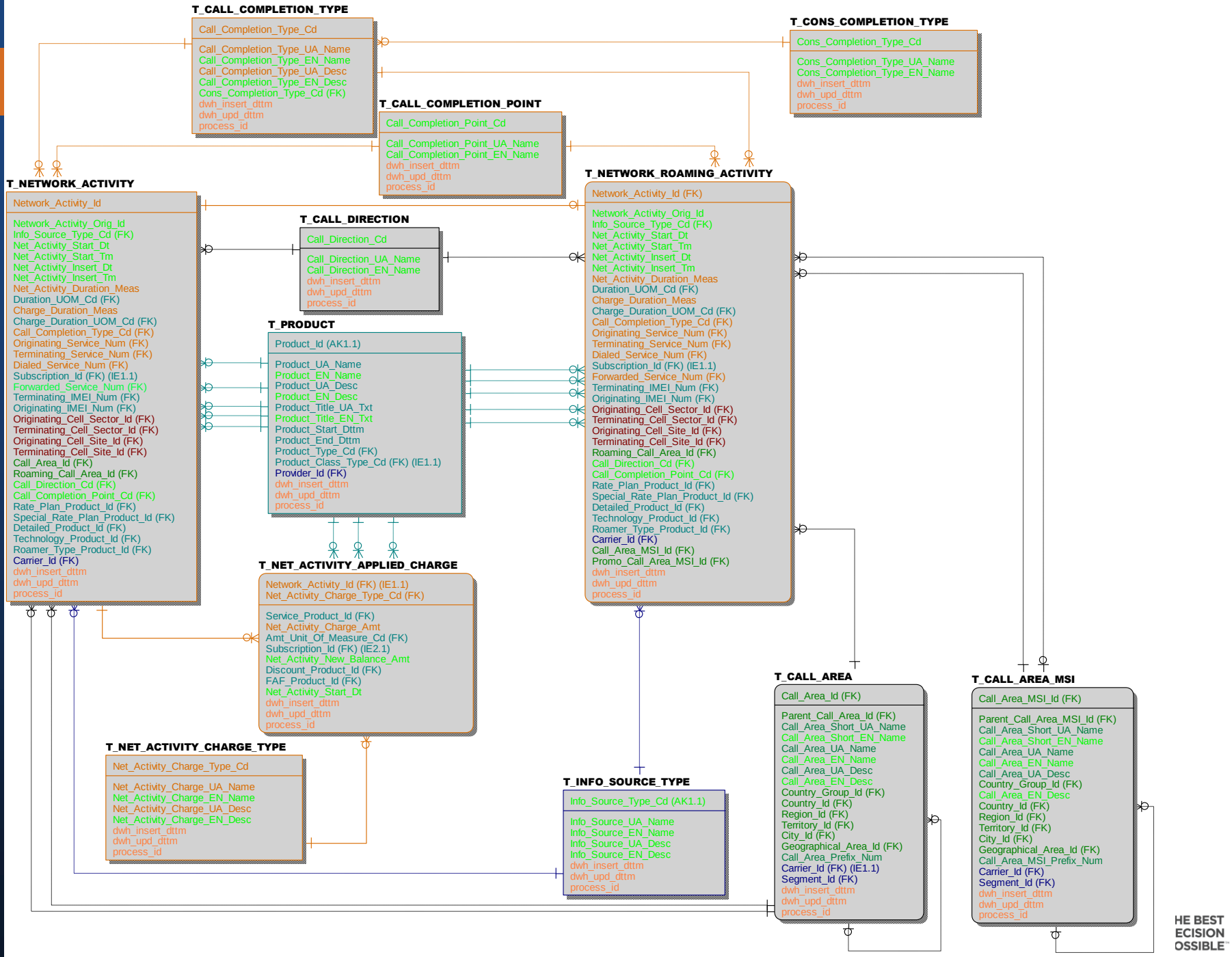




HE BEST
ECISION
OSSIBLE™

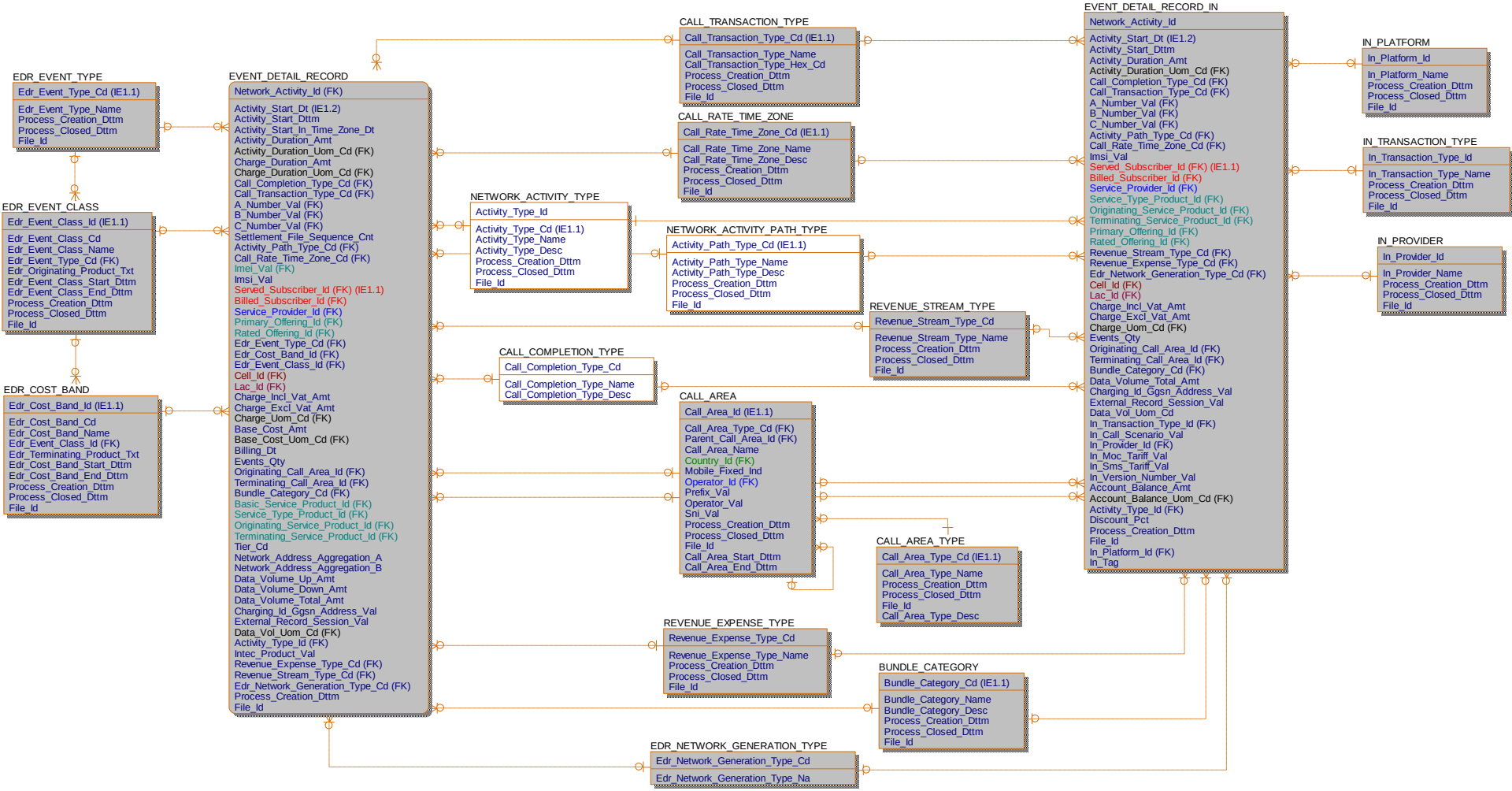
Network Activity – Super type roll down





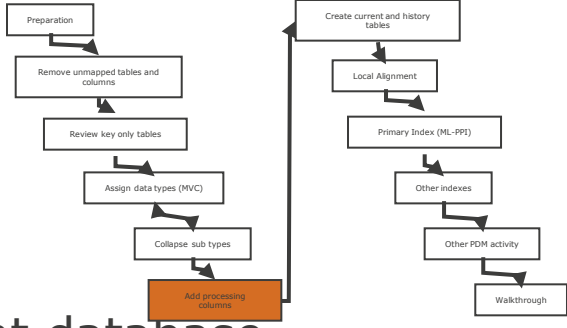
HE BEST
ECISION
OSSIBLE™

Super Type Roll Down customer example

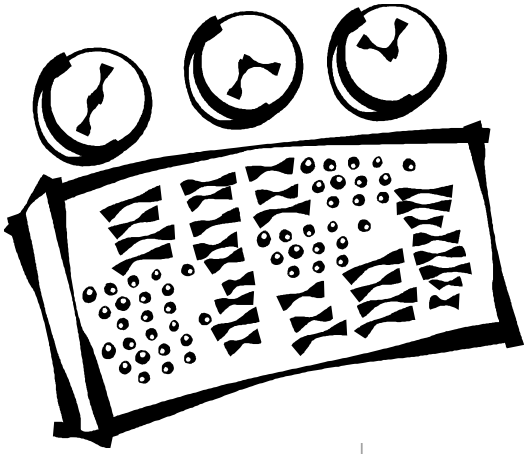


TERADATA. THE BEST DECISION POSSIBLE™

7. Add Processing Columns



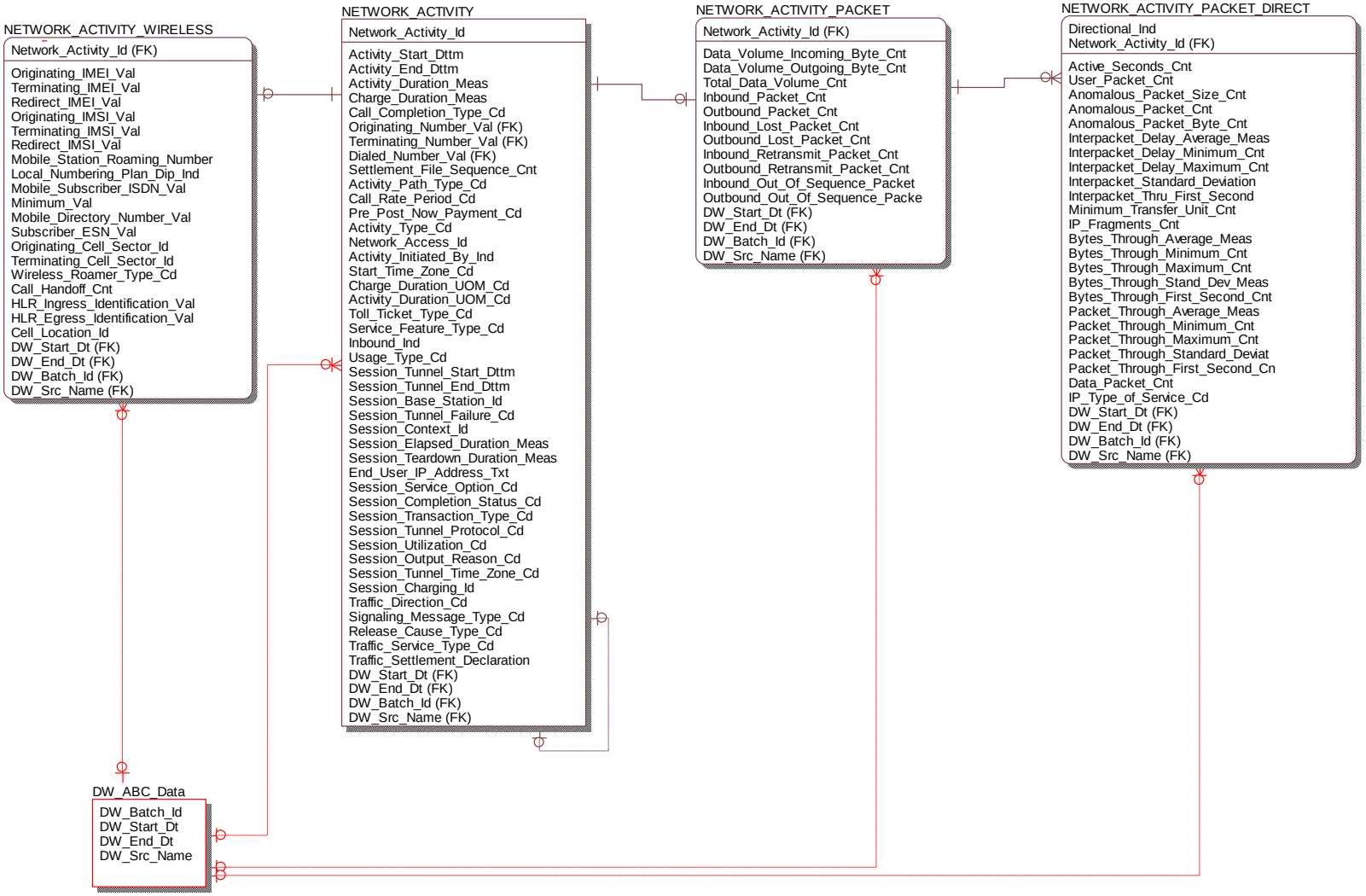
- Why
  - To support Audit and allow for true time variant database,
    - Adherence to Basel 2 and other legal compunction.
  - To support error recovery
  - Suggested reading 3 Tier Architecture Control Framework as a guide.
- Process
  - Review with client if they have any standards or needs outside normal EDW environment.
  - Make a standard set of columns.. Define domains carefully
  - Apply to EVERY table
  - Implement attributes to every table in standard form with standard domains



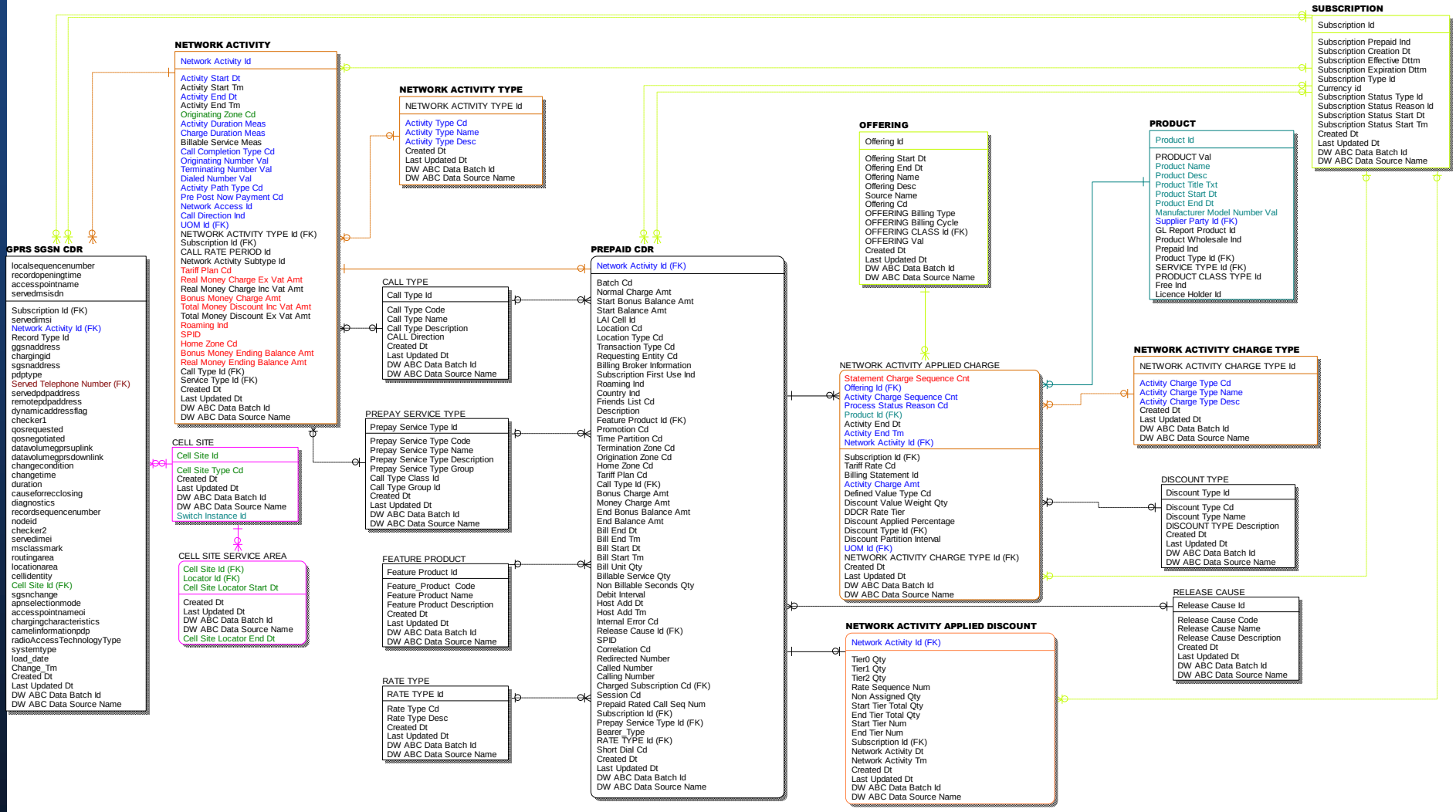
7. Add Processing Columns - Workshop

- The 3 Tier EDW Control Framework requires the following attributes be added to **EVERY** table
  - Start_Dt
  - End_Dt: compress HiDate
  - Record_Deleted_Flag: compress all
  - Source_System_Id: compress
  - Insert_Process_Id: compress
  - Update_Process_Id: compress
  - Insert_Proc_Name: compress
  - Update_Proc_Name: compress
- This is required for every table because it is easier and is consistent
- In the case of an insert only or transaction table the overhead is actually less than 11-13 bytes (9 if transaction style table)

Processing Columns: Network Activity



Processing Columns: Customer example



7. Focus on high impact tables: MVC

- Multi-value compression (MVC) is a Teradata-specific mechanism for reducing the stored size of a table WITHOUT incurring CPU costs during I/O operations.
- Why implement MVC in the first PDM? Because the database sizing during pre-sales will have assumed compression

Compression !

All sizes in GB							
Items	**Parameters**	**April-06**	**March-07**	**March-08**	**March-09**	**March-10**	**Comments**
1. Raw User Data (End of given period)		**1,625**	**2,851**	**4,407**	**6,431**	**9,061**	
2. Incremental Data added from existing systems	30%		1,080	1,404	1,825	2,373	
3. Historical Data added from new systems			380	-	-	-	
4. Incremental Data added from new systems	30%		101	262	341	443	
5. Estimated Purging (pa)	30%		335	110	143	186	
6. DW size (after compression and purging) (A)	20%	**1,625**	**2,281**	**3,526**	**5,144**	**7,249**	
7. Spool Area	40%	500	912	1,410	2,058	2,899	
8. DW size (A) + spool = (B)		**2,125**	**3,193**	**4,936**	**7,202**	**10,148**	
9. Raw data extensions (Indexes)	10%	-	228	353	514	725	
10. Summary and Applications (CM3, AML)	20%	175	456	705	1,029	1,450	
11. DW size (B) + RDE + Summary & Apps = (C)		**2,300**	**3,877**	**5,994**	**8,746**	**12,323**	
12. Individual User Data	30%	100	130	169	220	286	tempdb
13. ELT Processing/Staging Area		125	107	139	181	235	staging, utildb
14. DW size (C) + User + ELT processing = (D)		**2,525**	**4,114**	**6,302**	**9,146**	**12,843**	
15. System (DBC etc)		50	**200**	**225**	**250**	**300**	
16. Total database space = (D) + System		**2,575**	**4,314**	**6,527**	**9,396**	**13,143**	
17. RAID and FS OH	2.1		9,060	13,706	19,731	27,601	RAID-1 and TD OH
18. Total raw disk space		**6,968**	**9,060**	**13,706**	**19,731**	**27,601**	

If we do not implement MVC compression the data will outgrow the configuration too soon

7. Compression: VARCHAR

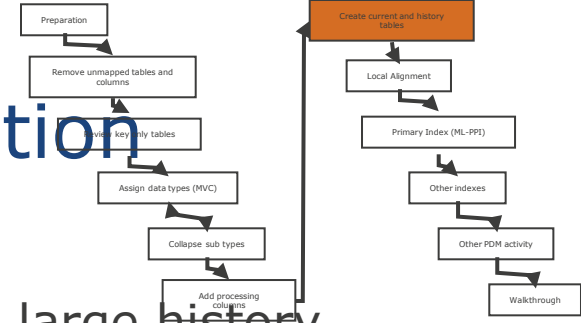
- Multi Value Compression cannot be applied to VARCHAR
  - > Convert most VARCHAR to fixed length CHAR and apply MVC
  - VARCHAR will generally be better when difference of maximum and average field length is high, and a high number of distinct values.
  - > Compression will generally be better when difference of maximum and average field length is low, and a low number of distinct values.

7. Focus on high impact tables: MVC

Other good reasons for implementing MVC

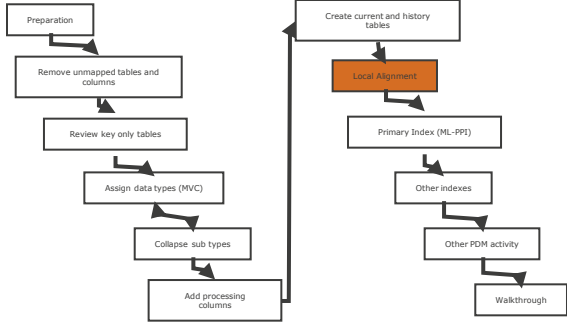
- Reducing row size reduces pages read to scan a table – performance improvement
- Frees up PermSpace for Spool, global temp tables etc – cumulative performance impact
- Provides contingency for data volume increases over initial estimate:
  - Sizing usually assumes approx 20% MVC saving
  - Experience shows usually achieve 30%
  - Seen documentation stating achieved 80% for CDR tables
  - Increases spool beyond expectation – good practice!
- Implementing "later" can provoke customer dissatisfaction;
  - "why have we bought more nodes than we *really* need?"

8. Current & History Tables Creation

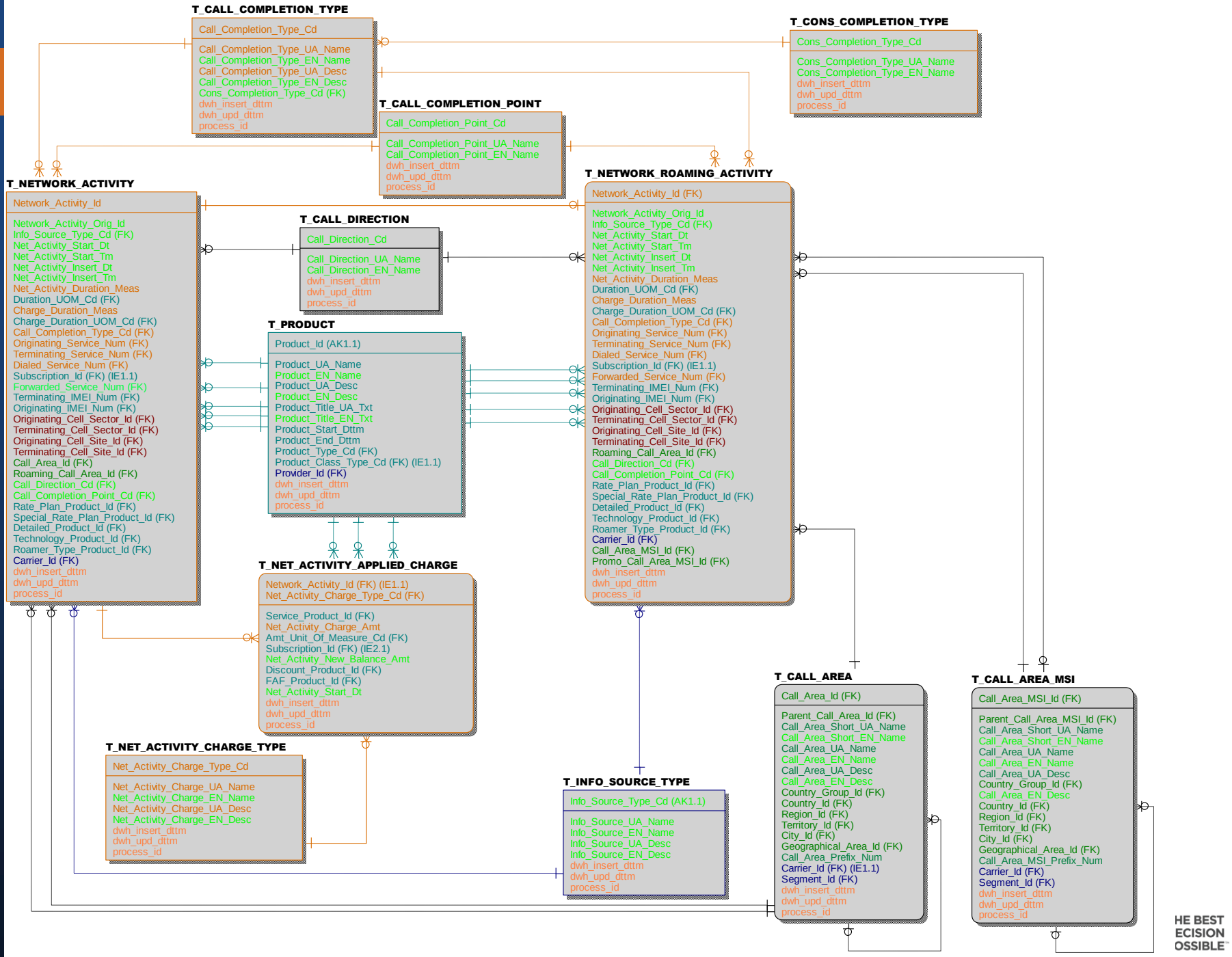


- Why
  - > High volume query on current state data from large history tables
- Process
  - > Obtain full usage metrics for history vs current
  - > If warranted then options exist
    - – Partition Primary Index
      - Queries must provide Partition value, but if so, better option
      - Multi Level Partitions for complex structures (TD12)
    - – New Table
      - Image of the current history table, needs program to populate
    - – Vertical partition
      - Divide the table up into parts, volatile and non volatile
      - Used and non used
    - – Join Index
      - Same concept as New Table but Teradata does the work
      - OR Same as Vertical Partition of table but Teradata does the work
      - May add a Current Table View support Join Index
  - > Document in Transformation Rules worksheet

9. Local Alignment

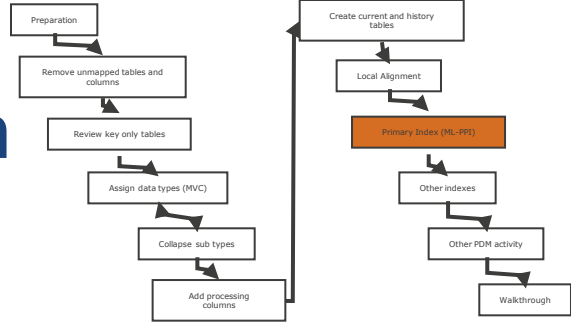


- Why
  - We must keep local standards and rules
  - For Active EDWs, the naming standards are vitally important to prevent confusion when supporting operational processes and systems
  - This is part of the DBA job and this process
- Process
  - Take all standards and update the column names and table names.
  - You may need to create a User Friendly set of 1:1 views to support the original customer LDM business style
  - Apply standards to domains of columns etc..
  - Change all names to the standards and add an entry into the Transformation Rules



HE BEST
ECISION
OSSIBLE™

10. Primary Index (PI) Validation



- Why
  - > Good query performance
    - ERwin and other tools that generate the draft physical schema will create all PI's base upon the Primary Key
    - Teradata can operate using those PI=PK situations, but it works **much** better with a properly aligned PI by Subject Areas. The concept you are after here is called Local AMP Join
    - Teradata exploits the hashed PI values for hash-merge joins (the most efficient join mechanism): this only works for tables with the same PI. No two customer LDM entities can have the same PK (normalisation rule)
    - Note the Teradata default for assigning a PI changes from TD13
    - Note that TD13 introduces non-PI tables to speed up load operations
  - > Good load performance
    - Transform and load to the PDM will require joins to existing PDM tables
    - Where the PIs are aligned, these are local AMP joins

10. Assign Primary Indices: rules of thumb

- Teradata tables have Primary Indexes
  - PIs do not necessarily uniquely identify column values
  - PIs are not necessarily (or even usually!) unique
- Good PI's have 3 characteristics
  - Rows likely to be joined have the same PI
  - Hash distribution based on the PI results in a non-skewed data distribution across AMPs
  - The PI is often specified in users' SQL – with unique values for ALL components of the PI (if not; full table scan)

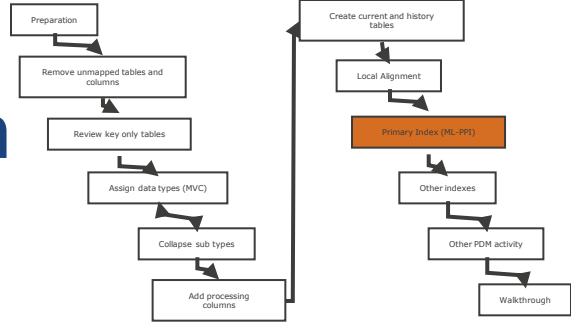
10. Surrogate vs 'natural' keys for PI's

- Advantages of Surrogate keys
  - Ensure uniqueness: data distribution
  - Independent of source systems
    - Re-numbering
    - Overlapping ranges
  - Database performance: best data type for PI's and joins
- Disadvantages of Surrogate keys
  - Have to allocate during ELT
  - Complex & expensive re-processing/data quality correction
  - Not used in queries – performance impact
  - Operational BI REQUIRES natural keys to join to operational systems
- Typically we use natural keys unless situation demands surrogates

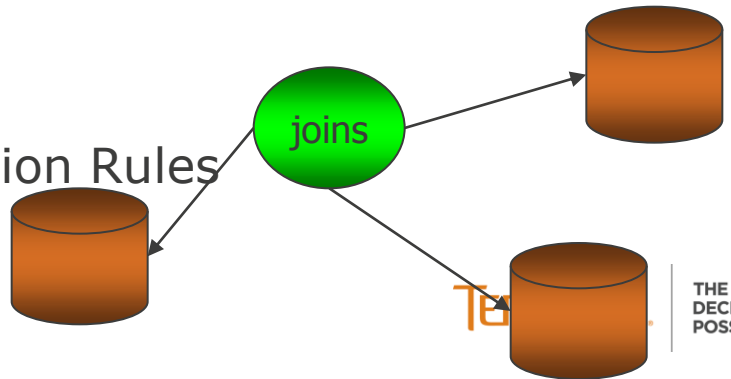
The PKs in the iLDMs are place holders for either natural or surrogate keys

See Orange Book: Physical Data Modeling for the Active Data Warehouse

10. Primary Index (PI) Validation



- Process
  - > Still need good data distribution.
  - > Review all entity PI's in the Subject Area for alignment from table to table.
  - > Check natural joins and see if the primary indexes can be aligned.
  - > Change PI's to allow for as many AMP local joins as possible.
  - > Verify that all selected PI's offer good data balance and verify no skewing is likely.
  - > Partitioned Primary Indexes (PPIs) should be considered for large tables and especially so if the table is bound to be queried along rolling dates (whether days, weeks or months). Before applying PPI review the specific sections of "Database Design" from Teradata.
  - > Verify cross Subject Area joins
  - > Add an entry into the Transformation Rules



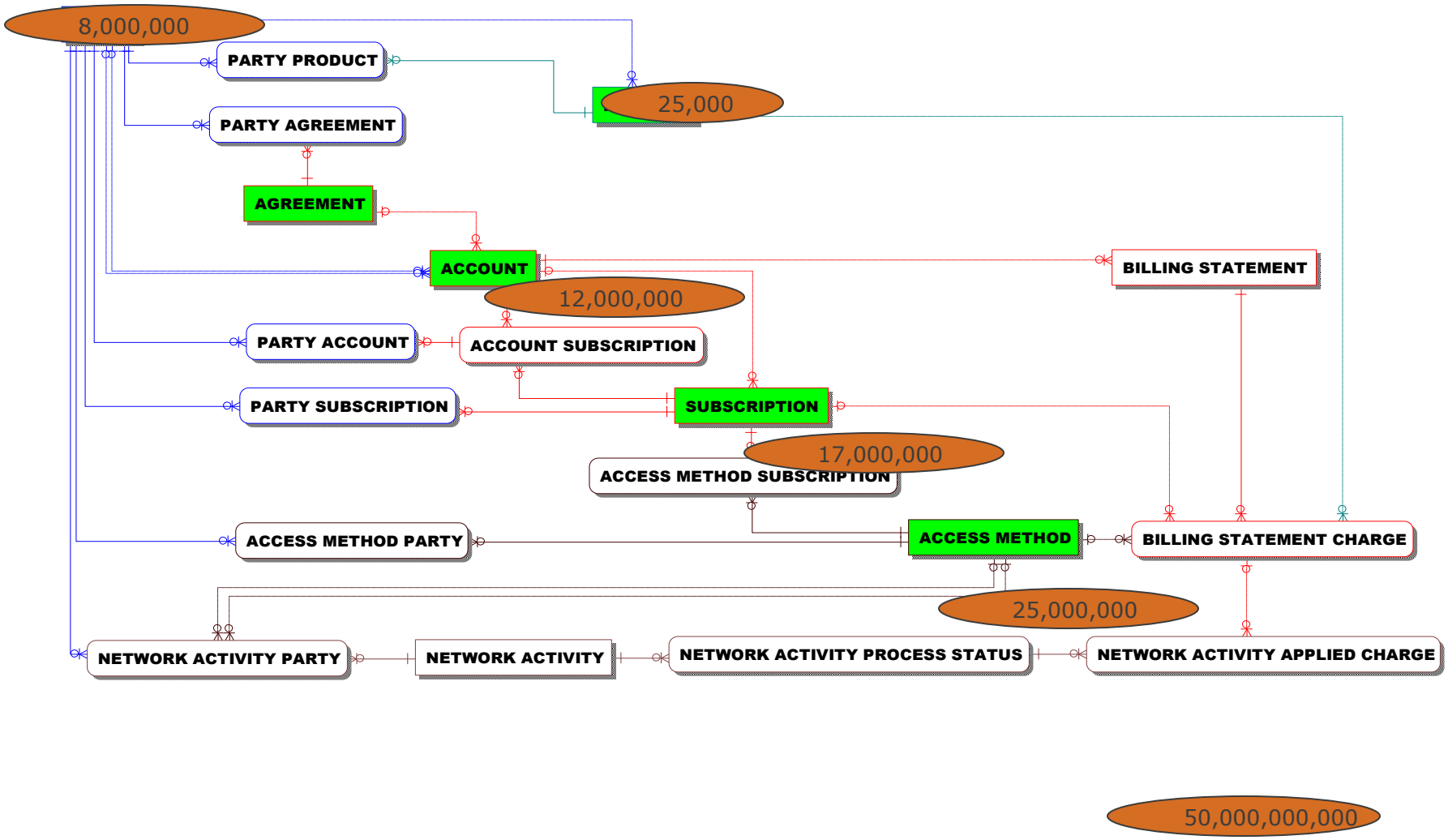
10. Partitioned Primary Index - general

- Why Implement PPI?
  - Many of the tables in the c-PDM are queried on date ranges
  - PPI does partition-elimination to reduce the physical I/O needed to return selected rows
  - PPI CAN be a major performance improvement
  - Backup can be by Partition (only backup the data that has changed)
  - Multi-temp can be by Partition
- Risk
  - Data Quality leads to skew to 1 or 2 partitions (eg NULL and 2999-12-31), or unable to partition at all
  - Minor performance risk to queries that ignore the partitioning criteria – they STILL do full table scan, but read-ahead can be impaired
- Process
  - Identify key tables that are queried by DATE range
  - Calculate optimum partitioning period: day, month, quarter
  - Test on representative data sample

10. Partitioned Primary Index: c-PDM

- Initial Candidates:
  - EVENT
  - NETWORK ACTIVITY
  - ACCT_BALANCE
  - SUBSCRIPTION_STATUS_HISTORY
- Review your scope and tables to identify other tables:
  - Queries have a range constraint on some column (especially, a date column) of the table
  - Queries have an equality constraint on some column of the table and that column is not the only primary index column or it is not a primary index column
  - Loaded periodically – typically daily
  - Where queries are date/time-constrained
    - This day this week, month, quarter vs this day last week, month, quarter
    - Compare SUM, AVERAGE, MAX, MIN, COUNT, etc this month vs another month
- Start point is PPI on most reliable BUSINESS DATE/DTTM

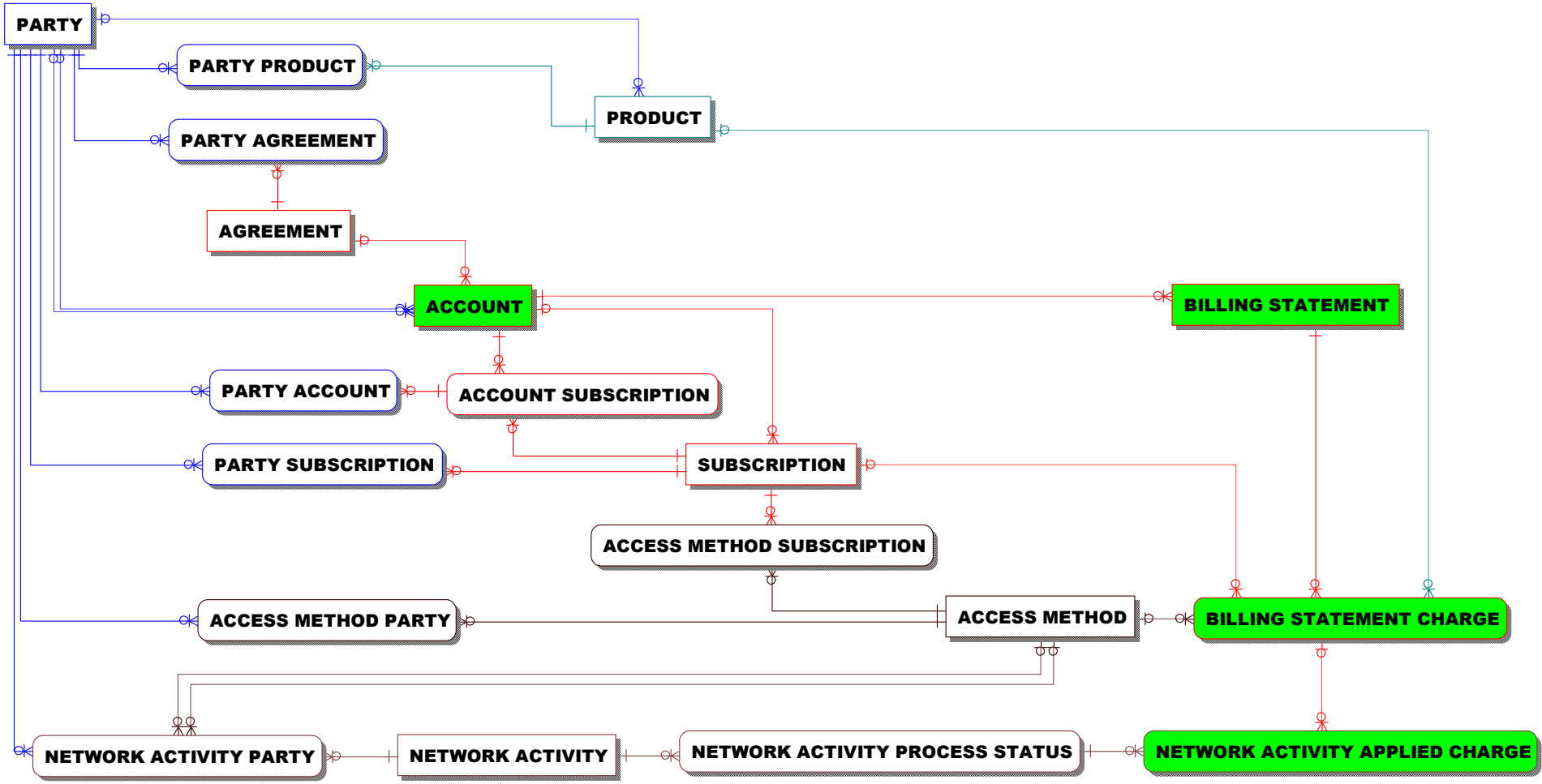
cLDM Entity row counts



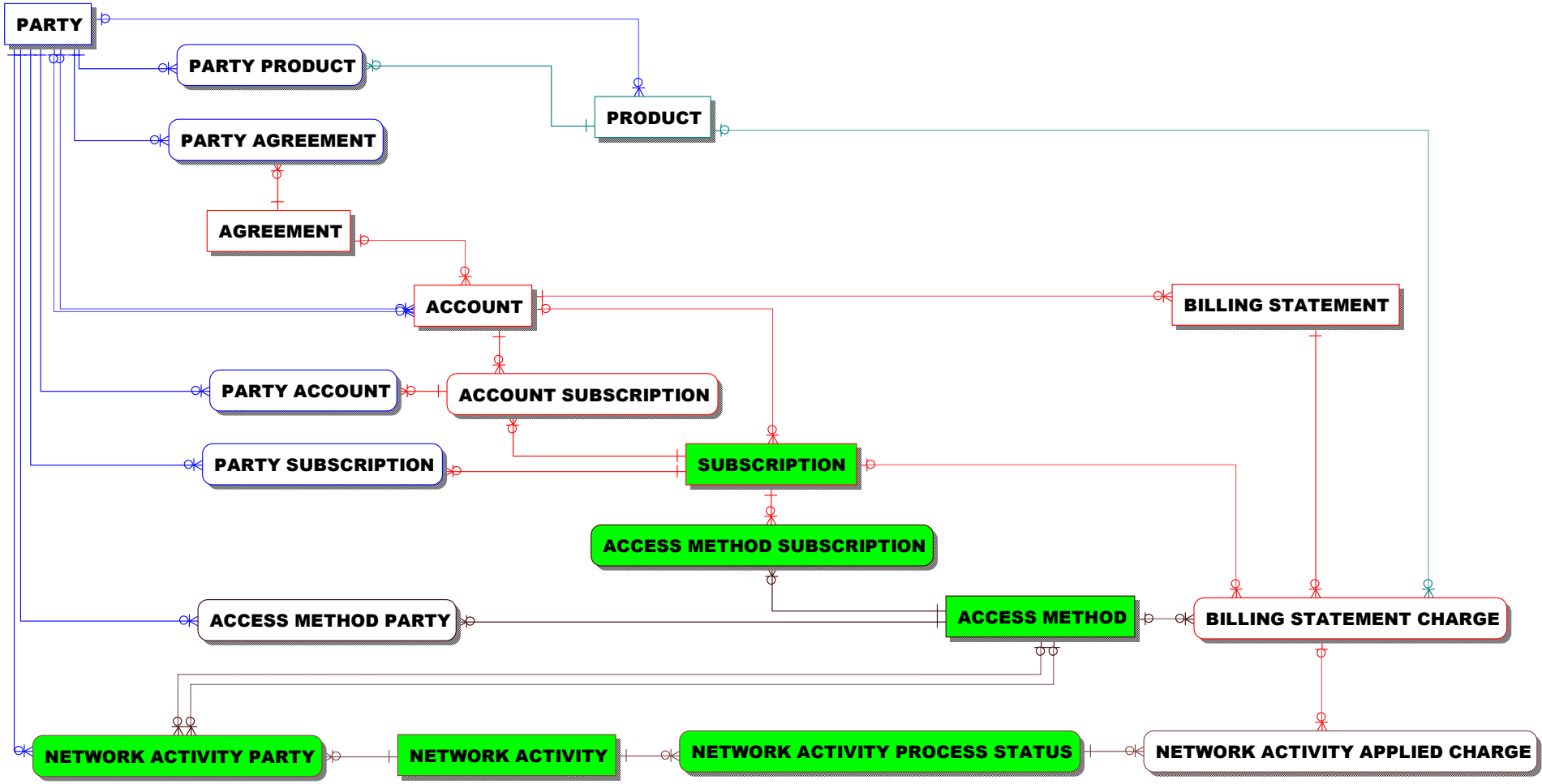
PI Selection
Business Question Analysis

Analysis Area	Analysis examples	Data Area Used
Acquisition Analysis	What is the average ARPU of the subscribers we acquire by channel, by region, by Service Provider, by month?	Segmentation, Subscription, Prepaid, Billing, Channel Hierarchy, Party, Geo-Location, Address, Network Activity Processing
Acquisition Analysis	What is the gross profit on prepaid customers younger than 24 years acquired through all franchised channels in the different Sales regions?	Party, Channel Hierarchy, Geo-Location, Address, Prepaid, Subscription, Network Activity Processing, Cost
Channel Preference Analysis	What is the channel distribution of high value customers i.e. what are the channels favored by the bulk of the high value customers?	Channel Hierarchy, Subscription, Network Activity Processing
Channel Preference Analysis	Which channels are experiencing the highest levels of churn amongst the Student Communicator segment?	Account, Subscription, Channel Hierarchy, Segmentation
Channel Preference Analysis	Which channel is favored by customers older than 55 years old who use the USB modem?	Product, Product Enrollment, Channel Hierarchy, Subscription, Party
Churn Drivers / Analysis / Prediction	Which channel, dealer and region is responsible for a disproportionate percentage of inactive prepaid connections relative to the size of their customer base and their monthly connections?	Party, Channel Hierarchy, Geo-Location, Address, Subscription
Churn Drivers / Analysis / Prediction	Provide all demographic, channel, recharge and monthly profiled usage data of MSISDNs who became inactive in a single flat table as input into the data mining software to determine the drivers for churn.	Party Demographic, Channel Hierarchy, Prepaid, Billing, Network Activity Base, Contract, Product Enrollment, Product
Churn Drivers / Analysis / Prediction	Of the contract customers who churned for preventable reasons, would their lifetime revenue contribution offset the investment to prevent them from churning??	Event, Subscription, Analytical Model, Billing, Cost
Churned Base Quality Analysis	What is the annualized churn rate in the prepaid base in each of the revenue bands (revenue bands in R5 increments)?	Network Activity Processing, Subscription
Churned Base Quality Analysis	What was the total lifetime value lost for all the corporate customers who churned and ported from STC in each of the regions in the previous month?	Event Number Portability, Geo-Location, Address, Subscription, Analytical Model, Network Activity Processing, Billing, Cost

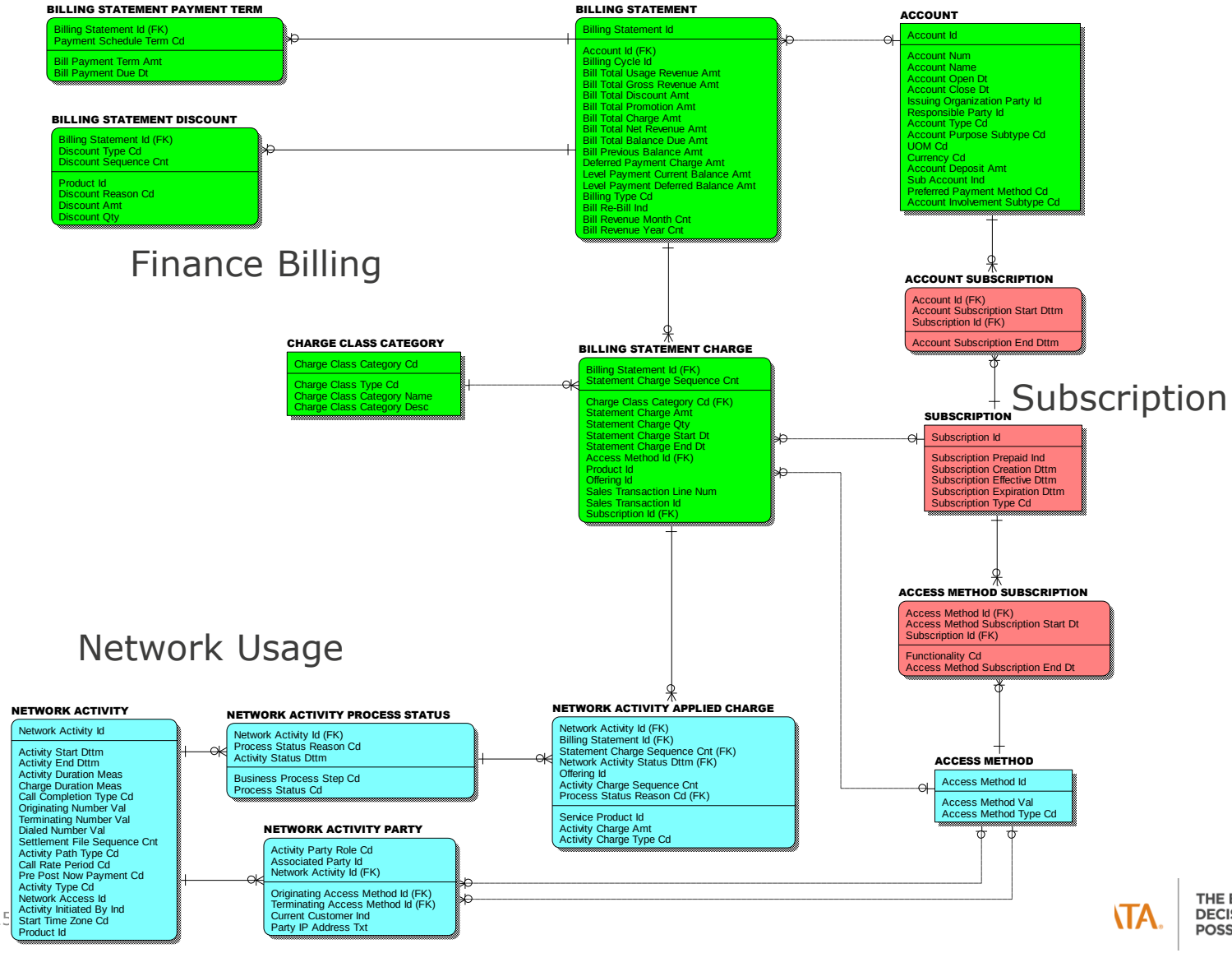
Account Access Path Analysis



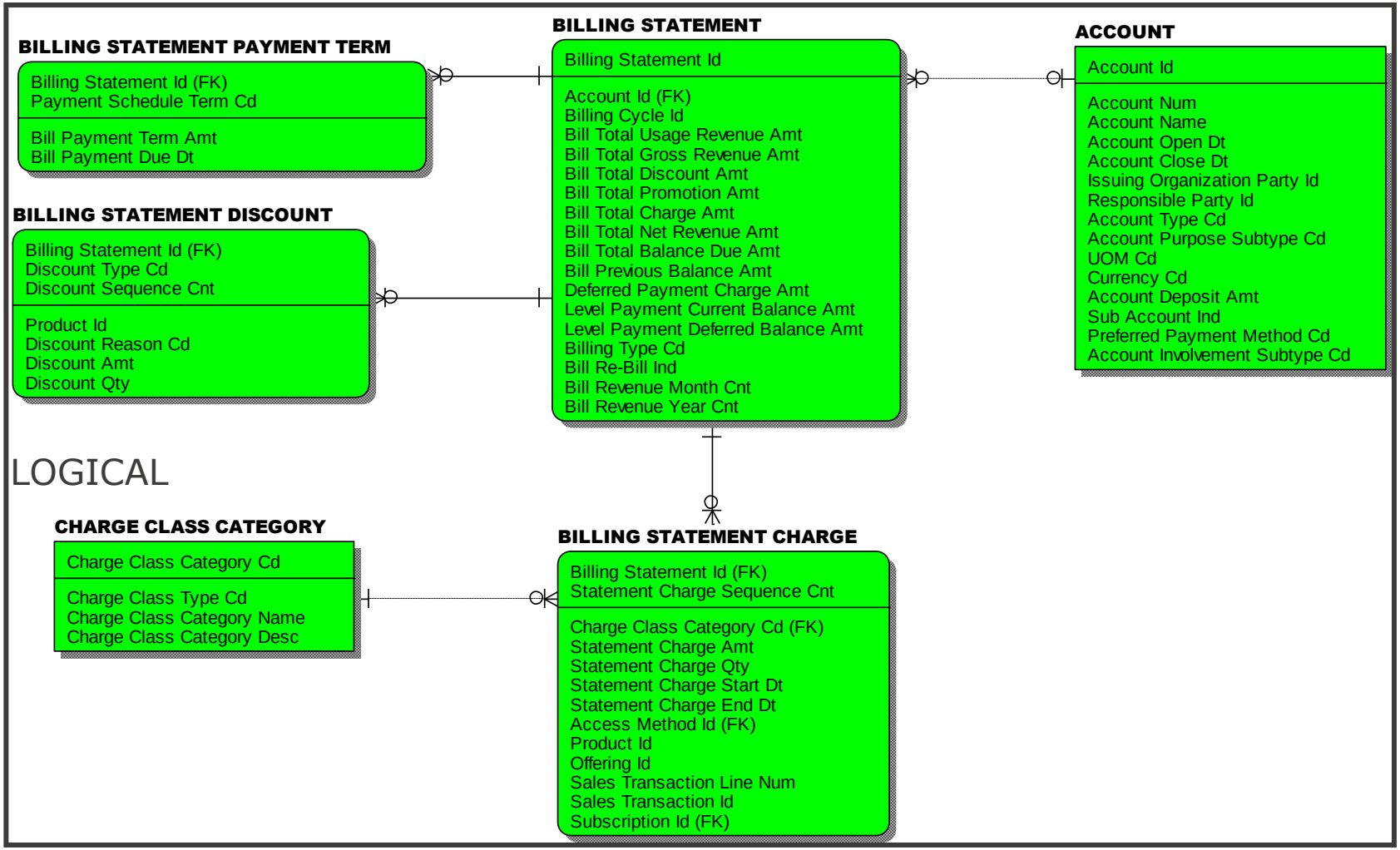
Subscription Based Access Path Analysis



Simple Example LDM



Review each subject area for potential Primary Index candidates: FINANCE

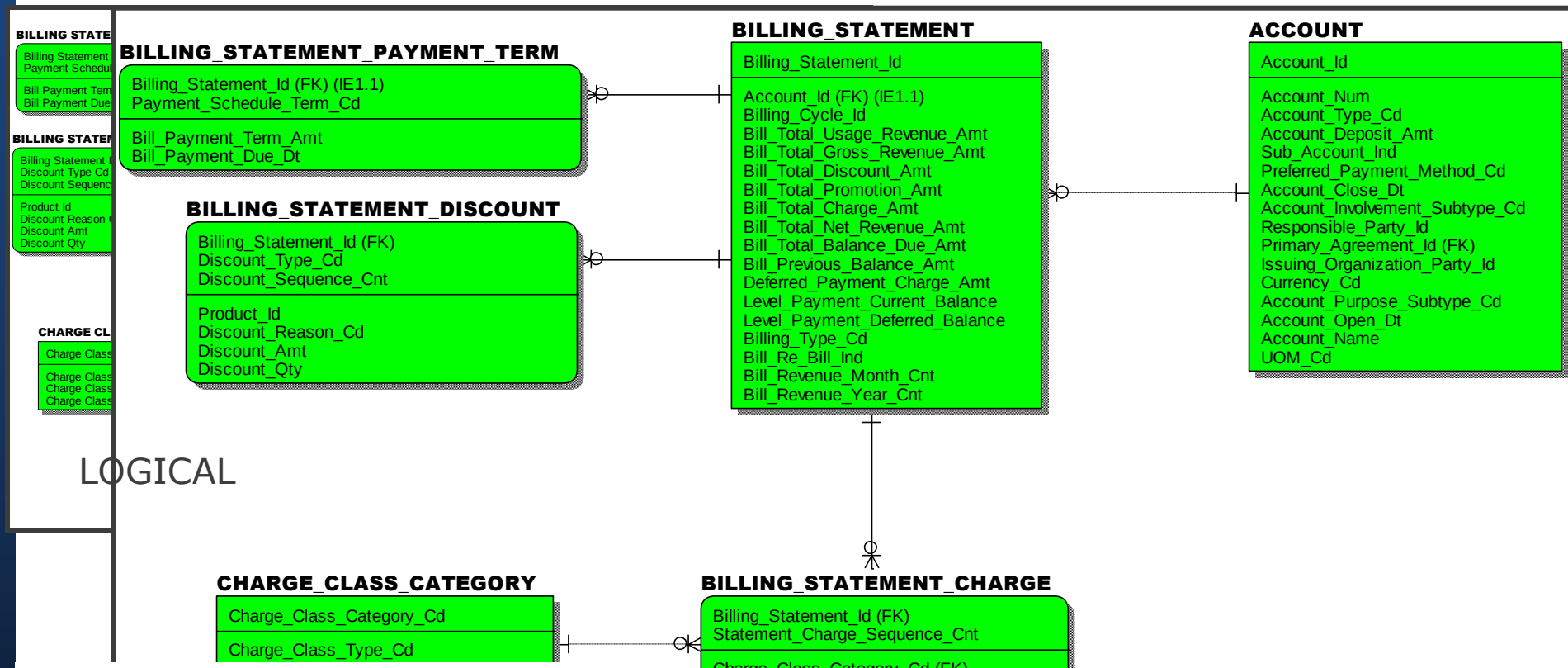


Review each subject area for PI candidates

Support Materials – customised iLDM, Business Questions and/or Reports, etc.

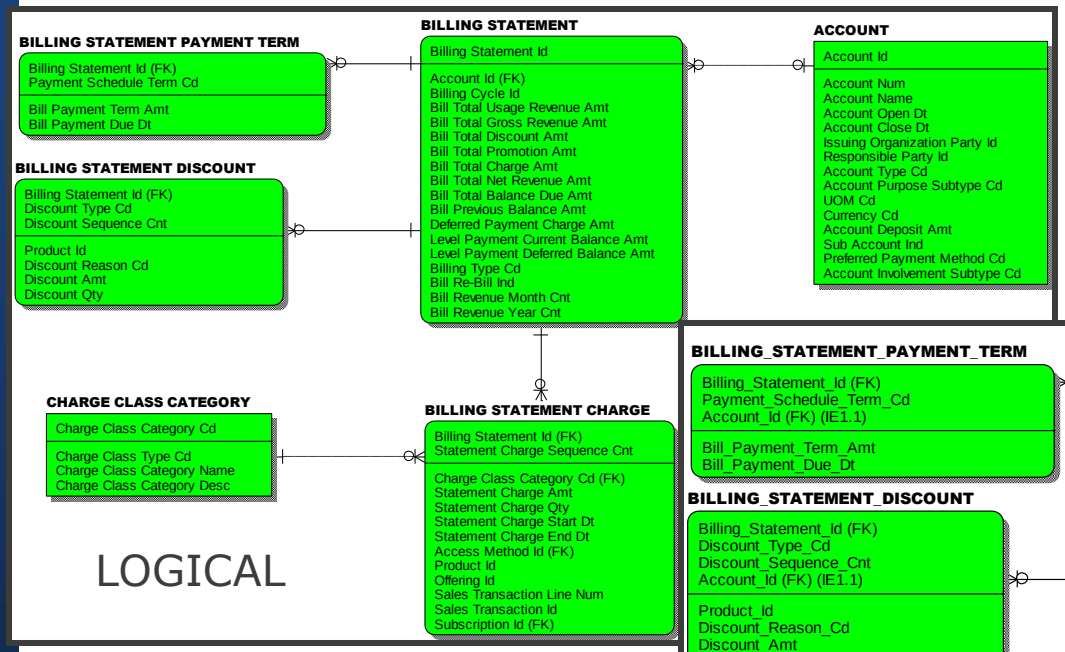
Likely candidate's will be selected from the Primary Key of the main entities in the subject area

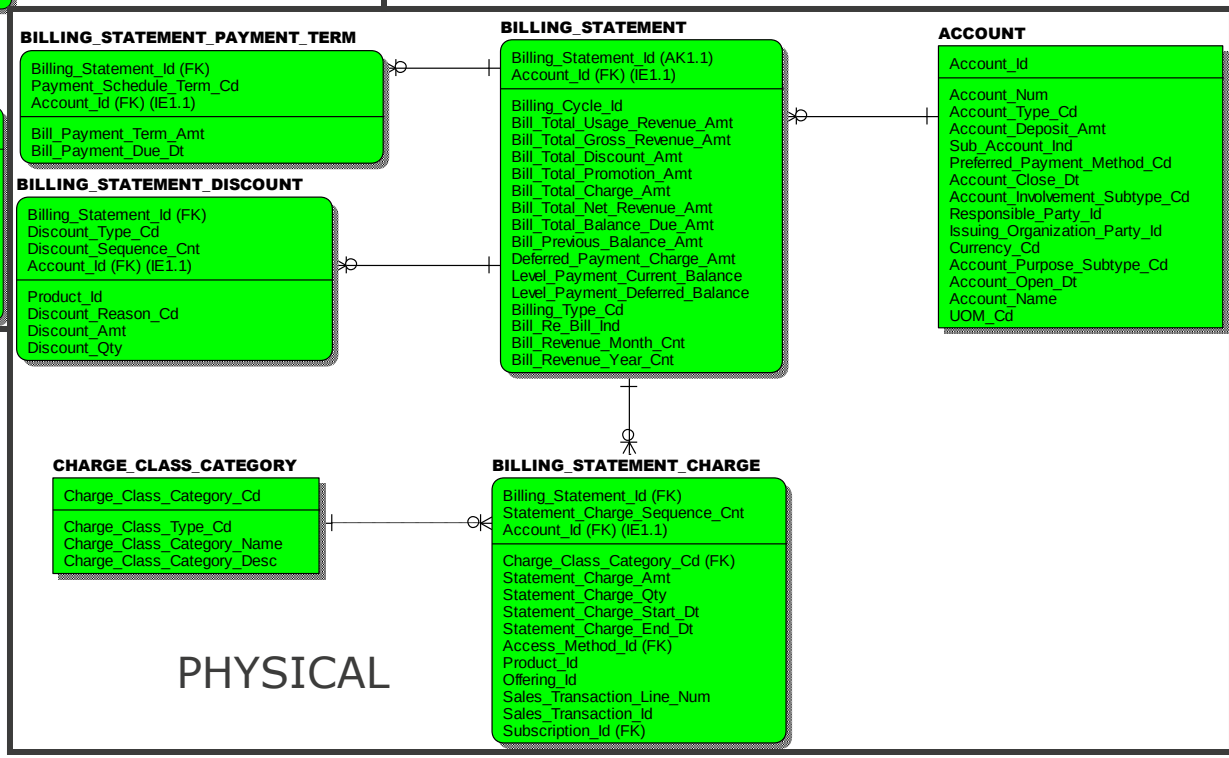
Select Account id as the PI of Bill Statement



- Number of times in BQ access paths
- Stability overtime
- Impact on Load Performance, Storage and latency
- Impact on Query Performance
- Check for Skew
- Reach of the Entity – number of used entities that can be reached

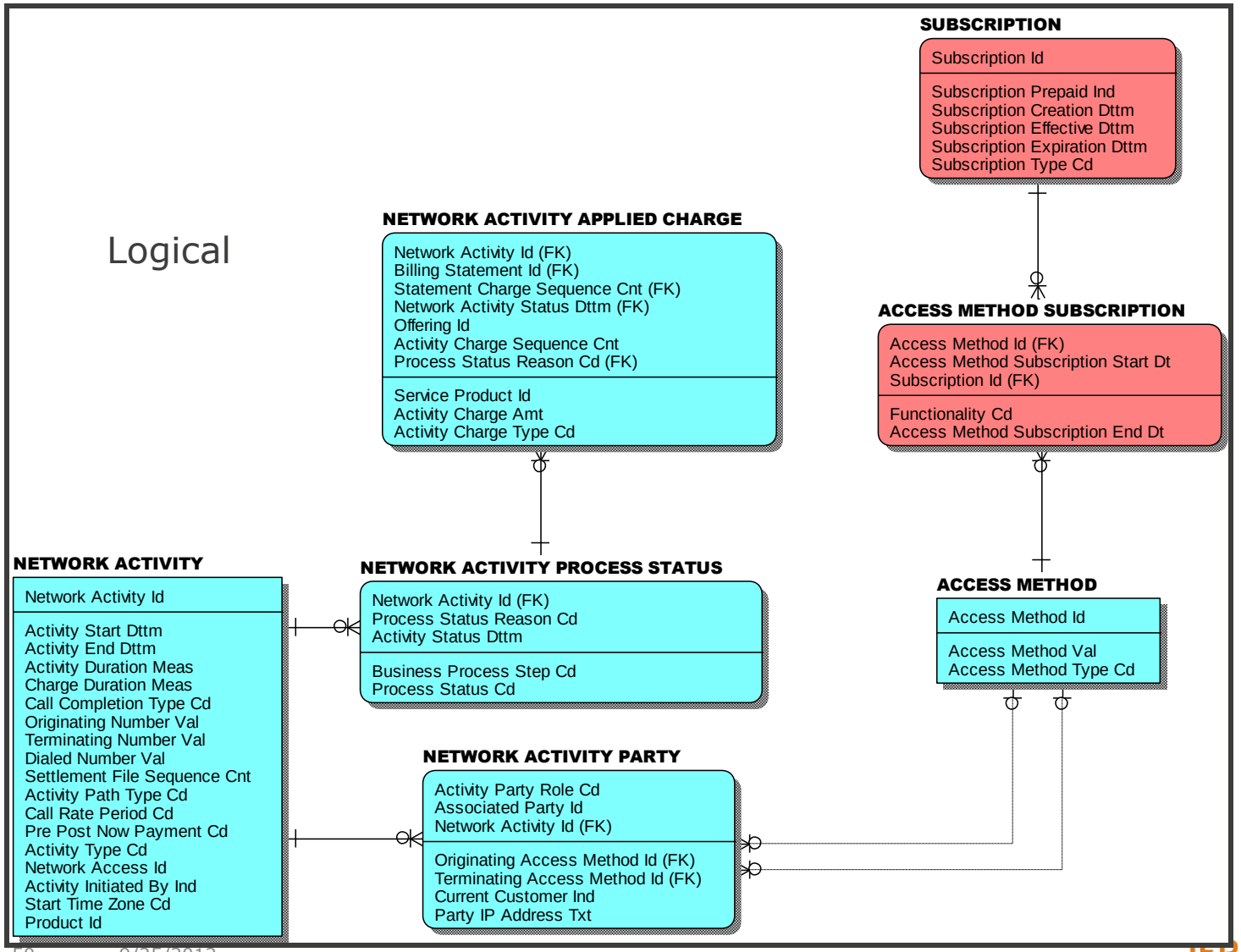
Select Account id as the PI & propagate





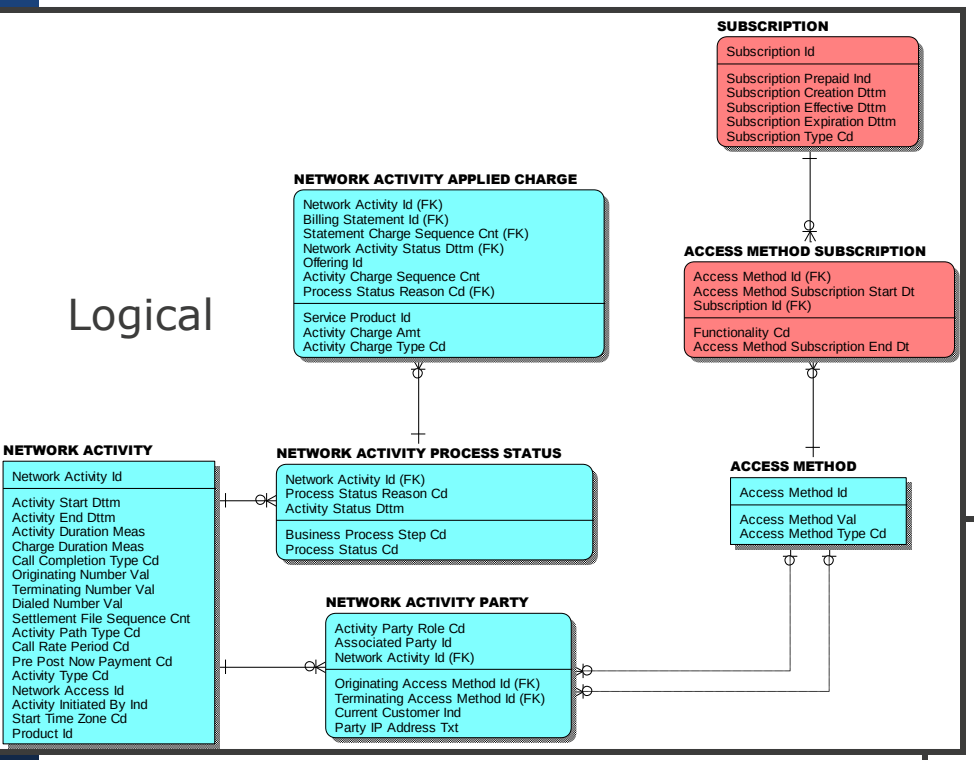
- Number of times in BQ access paths
- Stability overtime
- Impact on Load Performance, Storage and latency
- Impact on Query Performance
- Check for Skew
- Reach of the Entity – number of used entities that can be reached

NETWORK USEAGE & REVENUE Example



NETWORK USEAGE & REVENUE

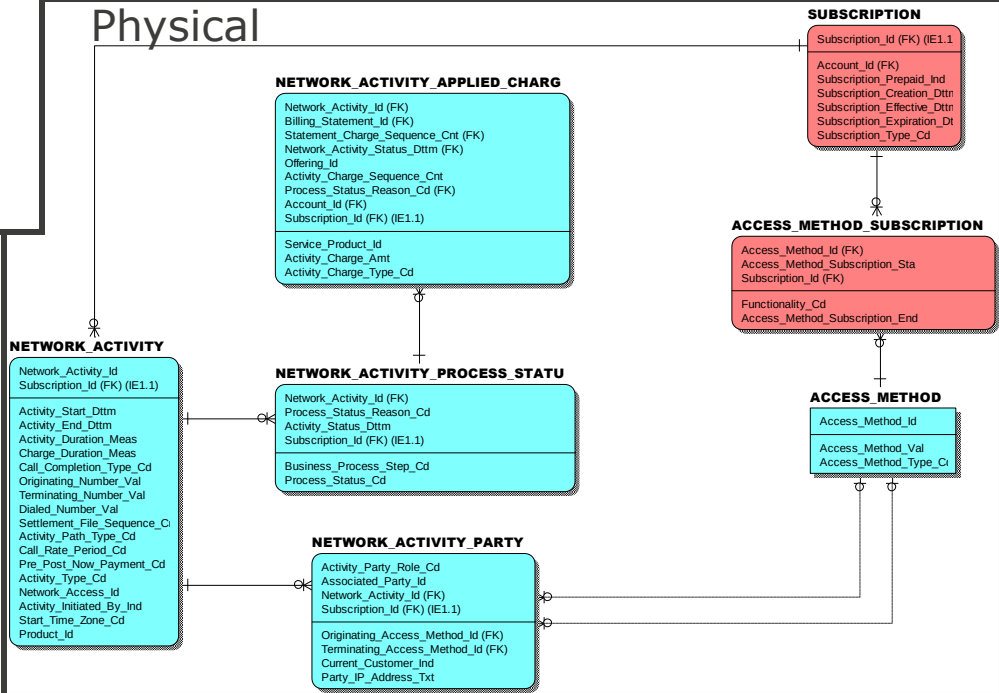
Logical



Subscription is the BQ preferred construct, created to be stable over time and gives excellent distribution.

A Business rule states that all users of the network must have a right to use.

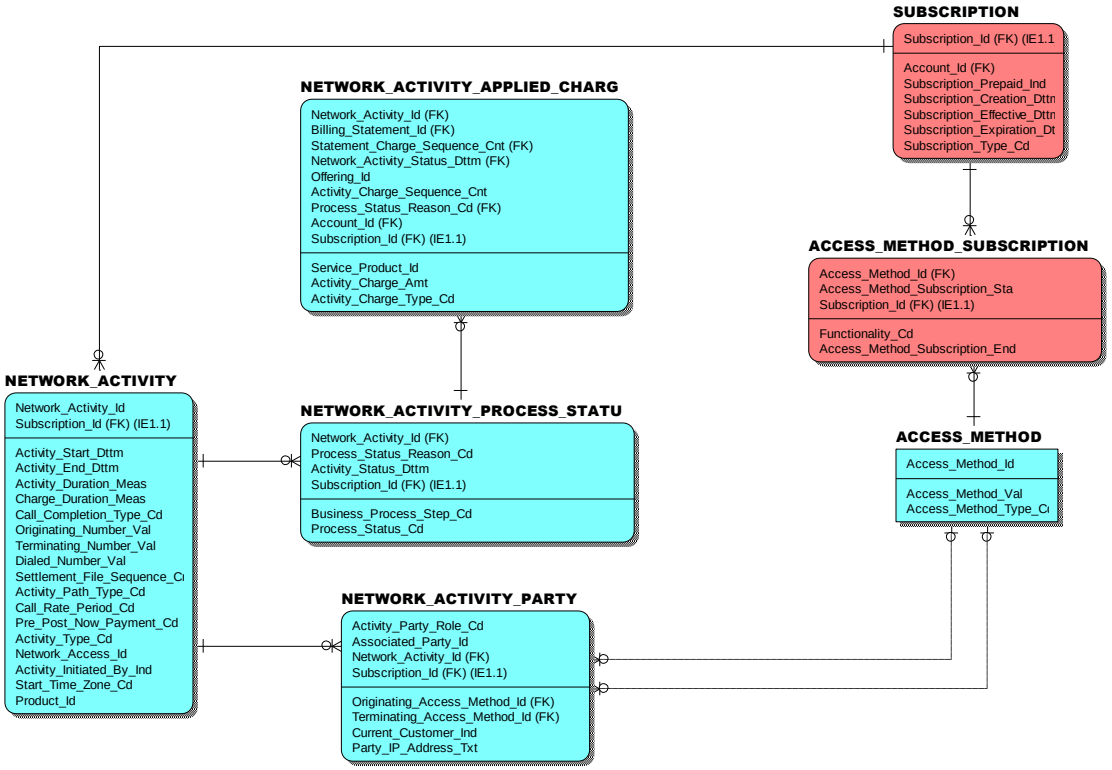
Physical



Subscription - Network Activity, Create Physical only Identifying Relationship

Subscription Id is an Alternate Index
Select Subscription Id as the NUPI

NETWORK USEAGE & REVENUE



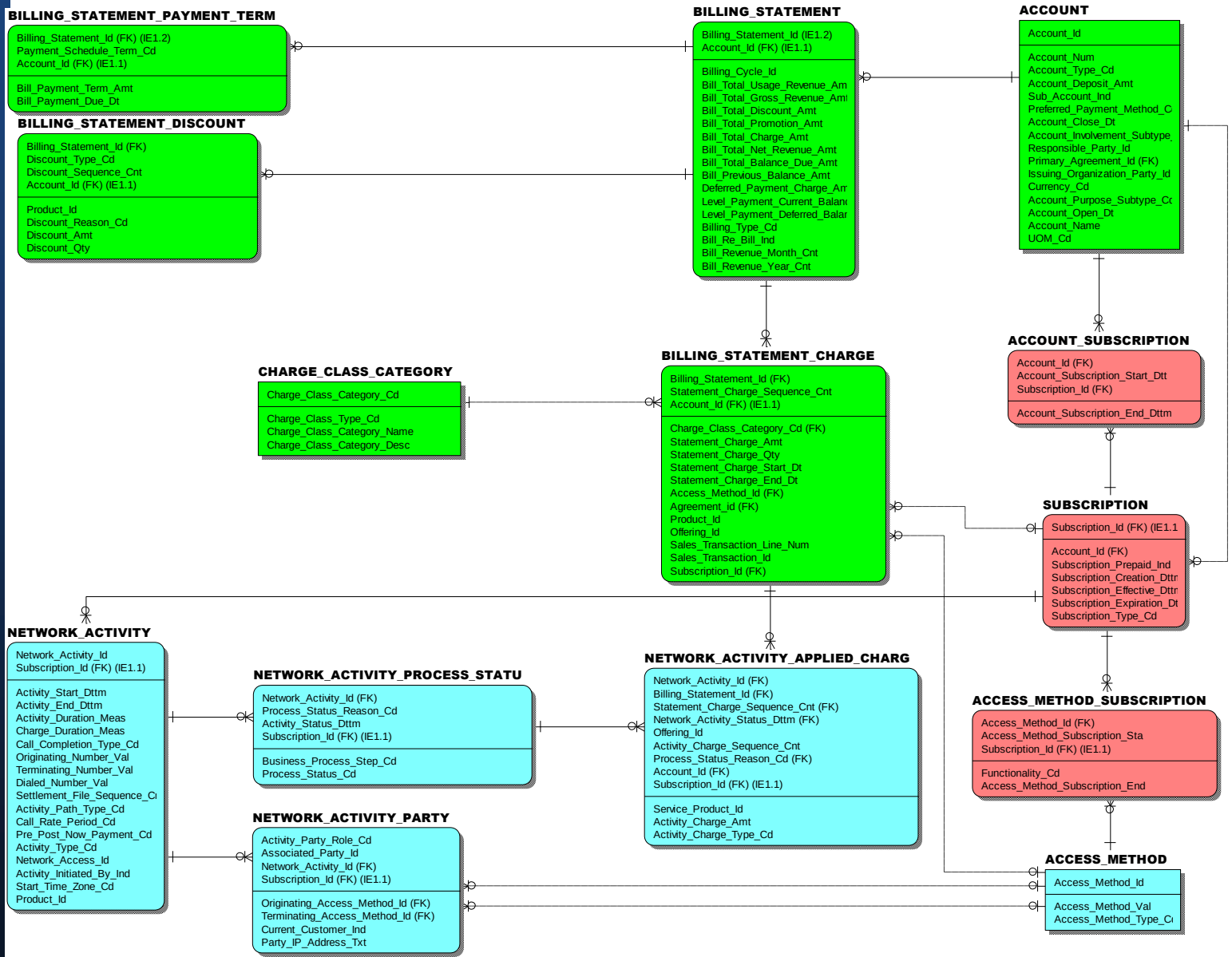
Now look at Access_Method_Subscription

Need for good distribution and access from both Access Method and Subscription?

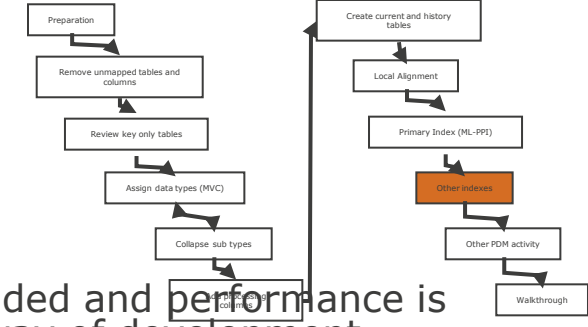
Access_Method_Subscription has a PI of Subscription_Id because although there are more Access_Methods than Subscription the BQ's use Subscription more and distribution is equally good.

Could also use a Join Index on Access_Method as it allows optimizer to pick best join method

Resulting physical design

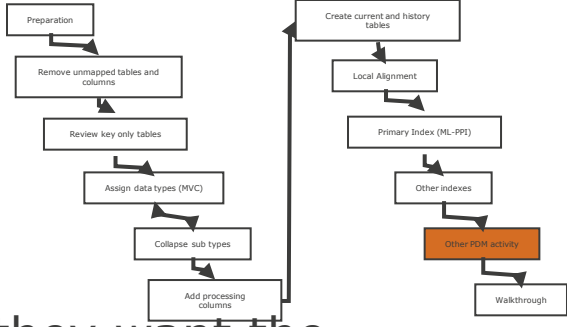


11. Other Index Considerations



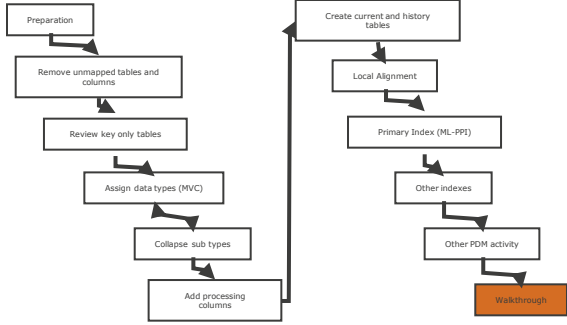
- Warning
  - Do **NOT** create any other indexes until after the data is loaded and performance is experienced. In general secondary indexes will get in the way of development
  - Extensive statistic gathering will assist the optimizer
  - Efficient plans based on good PI selection will often eliminate the need for other indexes. See Step 10.
- Why
  - Query Performance
- Process
  - See the white paper on Teradata Database Design
  - Review access paths for OLTP style access and determine if the Primary Indexes can support the requirement
  - Review any SLAs for both load and downstream
    - Secondary Indices slow loading
    - Some load utilities will not operate with certain secondary index choices
  - **Sparse** or **Join indexes** should **always be considered prior** to Secondary Indexes of the NUSI / USI kind
  - Balance overhead vs use
  - Add an entry into the Transformation Rules log

12. Other PDM Activity



- Why
  - Almost everyone has a wonderful idea of why they want the model changed once they start
  - Normally this happens some time during development because the requirement, source map or design task was not performed correctly
- Process
  - Check out the real reason of what is intended
  - Validate that against real requirements
  - Are you just being clever?
  - Try to avoid processing fixes in the DATABASE
  - Enforce Standards and Process

13. Walkthrough



- Why
  - > Being professional is about verification, checks and balances
  - > Part of being human is to make mistakes and miss things
  - > Opportunity to get alternative options or opinions
  - > The review document is part of the deliverable

  NOTE: EVERYONE at the Walkthrough now OWNS the PDM Deliverable!
- Process
  - > Standard Walkthrough Process
    - – Formal Walkthrough together as a group
    - – Subject Area by Subject Area
    - – Process by Process
    - – Worksheet entry by worksheet entry
  - > Recommended reviewers include the DBA, a programmer, a report developer, an architect and a business/logical modeler
  - > Inputs: customer LDM, New PDM, Worksheets
  - > Allow time to familiarize
- Objectives:
  - > Validate decisions to not do things
  - > Validate decisions implemented
  - > Mark errors or issues
  - > Approve or Reject

I have not presented rules…

- The ideas and process we have discussed are like the Pirate Code:

  "more guidelines than rules, don'tcha see?"

- All or NONE of these may apply in any design or implementation
- Treat the messages in this presentation as decision points – make auditable decisions!

Network Activity Customization

- Selective use of traffic and xDR types
  - Remove unwanted types
- Strong distinction between content transactions and messaging transactions
  - Create separate Messaging subtype
- No need to track xDR processing history
  - Remove NET ACTIVITY PROCESS STATUS
- Need to track processing history of all network activity attributes
  - Relate satellite "compartment" entities to NET ACTIVITY PROCESS STATUS