1. [TODO] [DEPENDS as needed] What should the “Blank String” be for the various Analytic fields, grab from this list when obvious:
   1. ‘None’
   2. ‘Unknown’
2. [TODO]Primary Family.  The GetFamily of Person (<https://github.com/SparkDevNetwork/Rock/blob/3ee719f7b68d273bf8a5b7ed2baf28ad00055ce7/Rock/Model/Person.cs#L2668>), isn’t guaranteed to give a consistent result in the cases of multiple active families.  I propose we add an “ORDER BY ID” so that it will be consistent. That way it would always be the “NEWEST” family record first (that’s what CCV’s datamart tables do).  Another solution is to introduce a “PrimaryFamilyId” to Person so that Rock no longer has to guess which family is their primary family (it would also speed up a lot of stuff).  It could default to whichever family they were added to first, but would be editable (sort of like GivingGroup). It might take a bit of work to figure out how we want to do this since a person can be moved from families, families can become inactive, removed, etc.  So, maybe we could go with guessing like we do for now and figure this out later.
   1. [THIS WAY]
      1. Sort so that [IsAdult] (Order so that Adult then Child), then by [OLDEST] (order by Group.Id ASC), wins.  In other words, if they are in two families, and they are an adult in one and a child in another, choose the family where they are an Adult.  If they are the same in both, choose the oldest family.
3. [TODO] [YES] Do we want AnalyticsDimPersonHistorical and AnalyticsDimPersonCurrent to be available to DataViews and Reports? I assume yes, but it would take some work to figure out how to deal with the Dynamic fields (Attributes)
   1. [YES] When you pick AttributeDimPerson for example, the field picker shows all the fields, including the attribute fields
   2. See <https://weblogs.asp.net/scottgu/dynamic-linq-part-1-using-the-linq-dynamic-query-library> for some possible ideas
   3. Random Notes
      1. Have a AttributeValuesXML column on AnalyticsDimPersonHistorical so that Attribute Values can be loaded efficiently from SQL without having to do a “One at a Time/Record By Record thingy”
      2. Have a “AnalyticsDynamicFields” Data Select that is smart enough to figure out how to create the Linq expression for SELECTs
      3. Have a “AnalyticsDynamicFields” Data Filter that is smart enough to figure out how to create the Linq expression for WHERE statements.
4. DimDate UI
   1. [Look for Mockup]
   2. Giving Month
      1. [] Use Sunday Date
         1. Hardcode don’t show as option ([X] Don’t cross Fiscal Years)
   3. Fiscal Year has a Month Picker
   4. Have two WeekDay columns WeekDay Int and WeekDayName string
   5. Have two GivingMonth columns GivingMonth Int and GivingMonthName string
5. Rock.Model.Attribute
   1. We had talked about adding an “IsAnalytic” as a field Rock.Model.Attribute,  but that would only be shown in the Attribute Config UI when the Model implement IAnalytic or IAnalyticHistorical.  I assume that would default to false for all existing Person/Family attributes? Or do we need to talk about that on a case by case (for the core ones) [YES, Jon will send me a list]
6. [TODO] Dim Family
   1. JON: How about instead Family, we do Group?, and then have a DimFamily\*\* views, etc
   2. CampusId (CampusName)
   3. Name (Group.Name or GetFamilyTitle?)
   4. Other fields?
      1. PrimaryHomeAddress ??
      2. HeadOfHouseholdPersonKey ??
   5. Attributes (Just like Person)

* When a new AttributeField is marked as IsAnalytic and IsAnalyticHistory, the next ETL will mark a bunch of records as History. For example
  1. Lets say that there are 100,000 people in the database, and 4000 of them have a FavoriteColor attribute that is NOT NULL
  2. Next, add FavoriteColor as IsAnalytic and IsAnalyticHistory
  3. Option #1) The next time the Person ETL runs, it will create a history record those 4000 people
  4. If we don’t like that behavior, we’ll have to figure out what we want to do instead
     1. Give the new FavoriteColor value to all of the HistoryRecords for that person..
        1. [THINK ABOUT, it would be cool if we could, but not if it’s a mess]…factoring in LastModifiedDateTime
           1. If the LastModifiedDateTime was 1/1/2013, create a HistoryRecord for that Person for that Date, based on the PreviousHistory record of the person
     2. Skip the “Mark As History” if the Field was just added in that ETL session?
     3. Factor in LastModifiedDateTime on the AttributeValue??