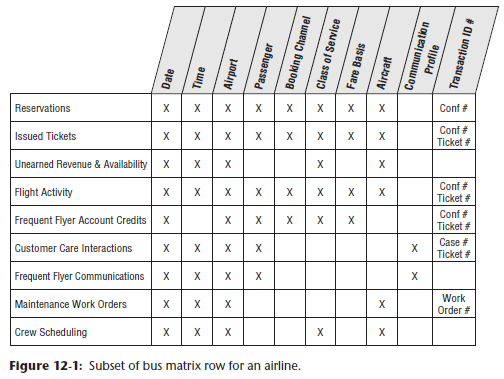
# Kimball Data Warehouse Toolkit

## Ch 12 – Transportation

* **Voyages** occur **whenever a person or thing travels from one point to another**, **perhaps w/ stops in the middle**
* Obviously, voyages are a fundamental concept for organizations in the travel industry
* **Shippers + internal logistical functions** also relate to the discussion, as well as **package delivery services + car rental companies**
* Somewhat unexpected, many transportation schemas are also applicable to telecommunications network route analyses (a phone network can be thought of as a map of possible voyages that a call makes between origin + destination phone numbers)
* We’ll draw on an airline case study to explore voyages + routes because many are familiar with the subject matter.
* The case study lends itself to a discussion of **multiple fact tables at different granularities**
* We’ll also elaborate on **dimension role playing** and additional date + time dimension considerations
* **Concepts:**
* Bus matrix snippet for an airline
* **Fact tables at different levels of granularity**
* Combining **correlated role-playing dimensions**
* Country-specific date dimensions
* Dates and times in multiple time zones
* Recap of localization issues

### Airline Case Study and Bus Matrix

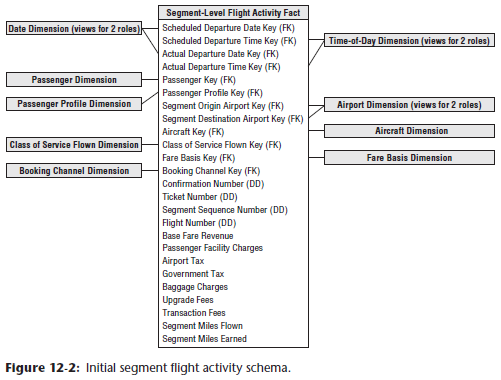
* We’ll begin by seeing a *simplified* bus matrix, + then dive into fact tables associated w/ flight activity
* The figure below shows a snippet of an airline’s bus matrix



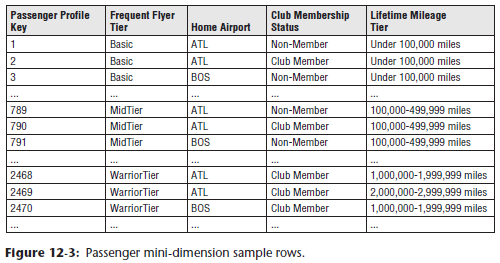
* This example includes an **additional column to capture the degenerate dimension Transaction ID #** associated with most of the bus process events
* **Like most organizations, airlines are keenly interested in revenue**
* In this industry, the **sale of a ticket represents *unearned* revenue**, + **revenue is *earned* when a passenger takes a flight** between origin and destination airports
* The business + DW/BI team reps decide the first deliverable should focus on **flight activity**
* The marketing department wants to analyze what flights the company’s frequent flyers take, what fare basis they pay, how often they upgrade, how they earn + redeem frequent flyer miles, whether they respond to special fare promotions, how long their overnight stays are, + what proportion of these frequent flyers have gold, platinum, aluminum, or titanium status
* The first project *doesn’t* focus on reservation or ticketing activity data that *didn’t* result in a passenger boarding a plane
* The DW/BI team will contend with those other sources of data in subsequent phases

#### Multiple Fact Table Granularities

* When it comes to the **grain** as you work through the 4-step design process, this case presents ***multiple* potential levels of fact table granularity, each having different associated metrics**
* At the ***most* granular level**, the airline captures data at the **leg level** 🡪 leg = an aircraft taking off at 1 airport and landing at another *without any intermediate stops*
* Capacity planning + flight scheduling analysts are interested in this *discrete* level of information because they can look at the number of seats to calculate **load factors by leg**
* Operational aircraft flight metrics are captured at the leg level 🡪 flight duration + the number of minutes late at departure + arrival
* Perhaps there’s even a dimension to easily identify on-time arrivals
* The ***next* level of granularity** corresponds to a **segment** = a single flight number (such as Delta flight number 40 or DL0040) flown by a single aircraft
* **Segments may have 1 or more legs associated with them**, but in most cases segments are composed of just 1 leg with a single take-off + landing
* If you take a flight from SF to Minneapolis with a stop in Denver with *no* aircraft or flight number change, you have flown 1 segment (SFO-MSP) but *two* legs (SFO-DEN and DEN-MSP)
* Conversely, if the flight flew nonstop from SF to Minneapolis, you would have flown 1 segment as well as 1 leg
* The **segment represents the line item on an airline ticket coupon**, + passenger **revenue + mileage credit is determined at the segment level**
* So, although some airline departments focus on leg level operations, the **marketing and revenue groups focus on segment-level metrics**
* Can also analyze flight activity by **trip**, which provides an accurate picture of **customer demand**
* In the prior example, assume the flights from SF to Minneapolis required the flyer to change aircraft in Denver
* In this case, the trip from SF to Minneapolis would entail 2 segments corresponding to the 2 involved aircraft
* In *reality*, the passenger just asked to go from SF to Minneapolis + the fact that she needs to stop in Denver is merely a necessary evil
* For this reason, **sales and marketing analysts are also interested in trip level data**
* Finally, the airline collects data for the **itinerary**, which is equivalent to the **entire airline ticket/reservation confirmation number**
* The DW/BI team + business reps decide to **begin at the segment-level grain**
* This **represents the lowest level of data with meaningful revenue metrics**
* Alternatively, *could* lean on the business for rules to allocate the segment-level metrics down to the leg, perhaps based on the mileage of each leg within the segment.
* The **DW inevitably will tackle the more granular leg level data** for the capacity planners + flight schedulers **at some future point**
* The **conforming dimensions** **built during this 1st iteration will be leveraged at that time**
* There will be **1 row in the fact table for each boarding pass collected from passengers**
* The **dimensionality associated with this data is quite extensive**, as illustrated below



* The **schema extensively uses the role-playing technique**.
* The **multiple date, time, + airport dimensions link to Views of a *single* underlying physical date, time, + airport dimension table**, respectively, as discussed originally in Chapter 6: Order Management
* The **passenger dimension** is a **garden variety customer dimension** with **rich attributes** captured about the most valuable frequent flyers
* Interestingly, frequent flyers are motivated to help maintain this dimension accurately because they want to ensure they’re receiving appropriate mileage credit
* *For a large airline, this dimension has tens to hundreds of millions of rows*
* Marketing wants to analyze activity by the frequent flyer tier, which can change during the course of a year
* In addition, you learned during the requirements process that users are interested in slicing + dicing based on flyers’ home airports, whether they belong to the airline’s airport club at the time of each flight, + their lifetime mileage tier
* **Given the change tracking requirements, coupled with the size of the passenger dimension, we opt to create a separate passenger profile mini-dimension**, as discussed in Chapter 5: Procurement, with **1 row for each unique combination of frequent flyer elite tier, home airport, club membership status, + lifetime mileage tier**
* Sample rows for this mini-dimension are illustrated below:



* You *considered* treating these attributes as type 2 SCD attributes, especially because the attributes don’t rapidly change
* But **given the number of passengers, you opt for a type 4 mini-dimension instead**
* Turns out marketing analysts often leverage this mini-dimension for analyses + reporting *without touching the millions of passenger dimension rows*
* The **aircraft dimension** contains information about each plane flown
* The **origin + destination airports** associated with each flight are **called out separately** to **simplify the user’s view of the data and make access more efficient**
* The **class of service flown** describes whether the passenger sat in economy, premium economy, business, or 1st class
* The **fare basis dimension** describes the terms surrounding the fare
* It would identify whether it’s an unrestricted fare, a 21-day advance purchase fare with change + cancellation penalties, or a 10% off fare due to a special promotion
* The **sales channel dimension** identifies how the ticket was purchased (through a travel agency, directly from the airline’s phone number, city ticket office, or website, or via another internet travel services provider, etc.)
* Although the sales channel relates to the *entire* ticket, **each segment should inherit ticket-level dimensionality**
* In addition, **several operational numbers** are associated with the flight activity data, including itinerary number, ticket number, flight number, + segment sequence number 🡪 **degenerate dimensions** **(DD)**
* The **facts captured at the segment level of granularity** include: base fare revenue, passenger facility charges, airport + government taxes, other ancillary charges + fees, segment miles flown, + segment miles awarded (in cases in which a minimum number of miles are awarded regardless of the flight distance)

#### Linking Segments into Trips

* Despite the powerful dimensional framework just designed, you cannot easily answer one of the most important questions about your frequent flyers, namely, “Where are they going?”
* The **segment grain masks the true nature of the trip**
* If you **fetch all the segments of a trip and sequence them by segment number, it is still nearly impossible to discern the trip start + endpoints, as most complete itineraries start + end at the same airport**
* If a lengthy stop were used as a criterion for a meaningful trip destination, it would **require extensive + tricky processing at the BI reporting layer whenever you try to summarize trips**
* The answer is to **introduce 2 more airport role-playing dimensions, trip origin + trip destination,** while **keeping the grain at the flight *segment* level**
* These are **determined during data extraction by looking on the ticket for any stop > 4 hours, the airline’s official definition of a stopover**
* **Need to exercise some caution when summarizing data by trip in this schema**
* **Some dimensions**, such as fare basis or class of service flown, ***don’t* apply at the *trip* level**
* On the other hand, it may be useful to see how many trips from SF to Minneapolis included an unrestricted fare on a segment
* In addition to **linking segments into trips** on the segment flight activity schema, if business users are constantly looking at information at the *trip* level, rather than by *segment*, you **might create an aggregate fact table at the trip grain**
* **Some** of the earlier **dimensions** discussed, such as class of service and fare basis, obviously **would *not* be applicable**
* Facts would include aggregated metrics like trip total base fare or trip total taxes, + additional facts that would appear only in this ***complementary* trip summary table**, such as number of segments in the trip
* However, **go to the trouble of creating this aggregate table *ONLY* if there were obvious performance or usability issues when you use the segment-level table as the basis for rolling up the same reports**
* If a typical trip consists of 3 segments, you **might barely see a 3X performance improvement with such an aggregate table, meaning it may not be worth the bother**

#### Related Fact Tables

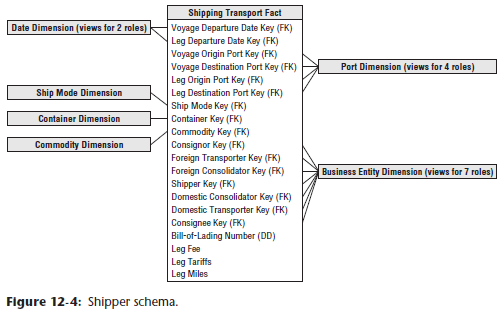
* As discussed earlier, you’d **likely create a *leg*-grained flight activity fact table to satisfy the more operational needs surrounding the departure + arrival of each flight**
* Metrics at the leg level might include actual + blocked flight durations, departure + arrival delays, + departure + arrival fuel weights.
* **In addition to flight activity, there will be related fact tables to capture reservations + issued tickets**
* Given the **focus on maximizing revenue**, there **might be a revenue + availability snapshot for each flight, which could provide snapshots for the final 90 days leading up to a flight departure with cumulative unearned revenue + remaining availability per class of service for each scheduled flight**
* This snapshot might include a dimension supporting the concept of “days prior to departure” to facilitate the comparison of similar flights at standard milestones, such as 60 days prior to scheduled departure

### Extensions to Other Industries

* Using the airline case study to illustrate a **voyage schema** makes intuitive sense because most people have boarded a plane at one time or another
* There are **several other variations on this theme**

#### Cargo Shipper

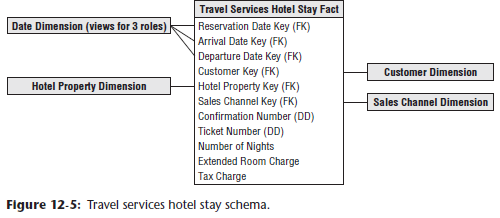
* The schema for a cargo shipper looks quite similar to the airline schemas just developed
* Suppose a transoceanic shipping company transports bulk goods in containers from foreign to domestic ports
* The items in the containers are shipped from an original shipper to a final consignor, the trip can have multiple stops at intermediate port, it’s possible containers may be off -loaded from one ship to another at a port, + it’s possible one or more of the legs may be by truck rather than ship
* As illustrated below, the **grain of *this* fact table is the container on a specific bill-of-lading number on a particular leg of its trip**



* The **ship mode dimension** identifies the type of shipping company and specific vessel
* The **container dimension** describes the size of the container + whether it requires electrical power or refrigeration
* The **commodity dimension** describes the item in the container
* Almost anything that can be shipped can be described by **harmonized commodity codes**, which are a kind of **master conformed dimension** used by agencies, including U.S. Customs
* The consignor, foreign transporter, foreign consolidator, shipper, domestic consolidator, domestic transporter, + consignee are all **roles** played by a **master business entity dimension** that contains all possible business parties associated with a voyage
* The **bill-of-lading number is a degenerate dimension**
* Also, we assume the fees + tariff s are applicable to the individual leg of the voyage.

#### Travel Services

* If you work for a travel services company, you **can complement the flight activity schema with fact tables to track associated hotel stays and rental car usage**
* **These schemas would share several common dimensions**, such as the date and customer.
* For hotel stays, the grain of the fact table is the *entire* stay, as illustrated below



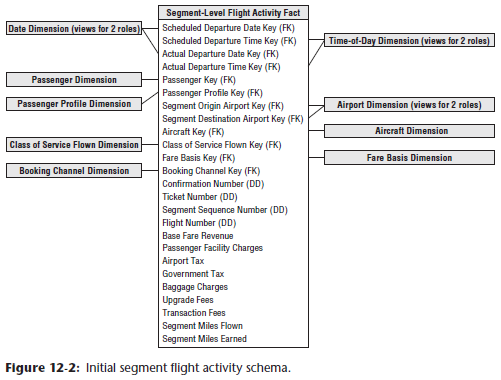
* The grain of a similar car rental fact table would be the entire rental episode.
* *Of course, if constructing a fact table for a hotel chain rather than a travel services company, the schema would be much more robust because you’d know far more about the hotel property characteristics, the guest’s use of services, and associated detailed charges*

### Combining Correlated Dimensions

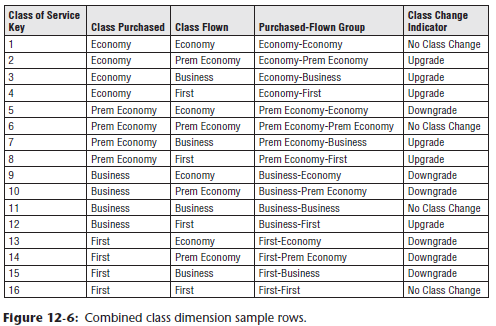
* It was stated previously that **if a many-to-many relationship exists between 2 groups of dimension attributes, they should be modeled as *separate* dimensions with *separate* FKs in the fact table**
* ***Sometimes*, however, you encounter situations where these dimensions can be *combined* into a single dimension rather than treating them like that**

#### Class of Service

* The current schema draft includes the **class of service flown dimension**



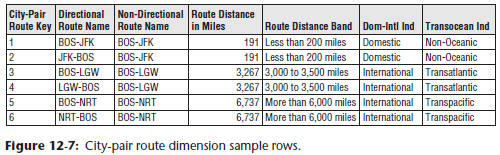
* Following a design checkpoint with the business community, you learn business **users *also* want to analyze the *booking* class *purchased***
* In addition, they want to easily filter + report on activity based on whether an upgrade or downgrade occurred
* Your *initial reaction might be to include a 2nd role-playing dimension and FK in the fact table to support both the purchased + flown class of service*
* In addition, you’d need a 3rd FK for an upgrade indicator, as otherwise, the BI application would need to include logic to identify numerous scenarios as upgrades, including economy to premium economy, economy to business, economy to 1st, premium economy to business, + etc.
* In *this* situation, however, there are only 4 rows in the class dimension table to indicate 1st, business, premium economy, + economy classes
* Likewise, the upgrade indicator dimension also would have just 3 rows in it, corresponding to upgrade, downgrade, or no class change
* **Because the row counts are so small, you can elect instead to *combine* the dimensions into a *single* class of service dimension**, as illustrated below



* The Cartesian product of the separate class dimensions results in a 16-row dimension table (4 class purchased rows times 4 class flown rows)
* You also have the opportunity in this combined dimension to describe the relationship between the purchased + flown classes, such as a class change indicator
* **Think of this combined class of service dimension as a type of junk dimension** (Chapter 6)
* i.e., **Combines together a number of miscellaneous, low-cardinality ﬂags + indicators, rather than making separate dimensions for each ﬂag + attribute**
* In *this* case study, the attributes are tightly correlated
* Other airline fact tables, such as inventory availability or ticket purchases, would invariably reference a conformed class dimension table with just 4 rows.
* **NOTE: In most cases, role-playing dimensions should be treated as separate logical dimensions created via views on a single physical table, but in isolated situations, it may make sense to combine the separate dimensions into a single dimension**
* **Notably when the data volumes are extremely small *or* there is a need for additional attributes that depend on the combined underlying roles for context and meaning**

#### Origin and Destination

* Likewise, **consider the pros and cons of combining the origin and destination airport dimensions**
* In this situation, **the data volumes are more significant, so separate role-playing origin + destination dimensions seem more practical**
* ***However*, users may need additional attributes that depending on a *combo* of origin + destination**
* **In addition to accessing the characteristics of each airport, business users also want to analyze flight activity data** by the distance between the city-pair airports, as well as the type of city pair (such as domestic or trans-Atlantic)
* *Even a seemingly simple question regarding the total activity between SF (SFO) and Denver (DEN), regardless of whether the flights originated in SFO or DEN, presents some challenges with separate origin and destination dimensions*
* **SQL experts could surely answer the question programmatically with separate airport dimensions, but what about the less empowered?**
* And even if experts *can* derive the correct answer, there’s no standard label for non-directional city-pair routes
* Reporting applications may label it SFO-DEN, DEN-SFO, San Fran-Denver, or Den-SF, + so on
* **Rather than embedding inconsistent labels in BI reporting application code, attribute values should be stored in a dimension table, so common standardized labels can be used throughout the organization**
* It’d be a shame to go to the bother of creating a DW + then allowing application code to implement inconsistent reporting labels, + business sponsors of the DW/BI system won’t tolerate that for long.
* To satisfy the need to access additional city-pair route attributes, you have 2 options
* **1) Merely to add another dimension to the fact table** for the city-pair route descriptors, including directional route name, non-directional route name, type, + distance, as shown below



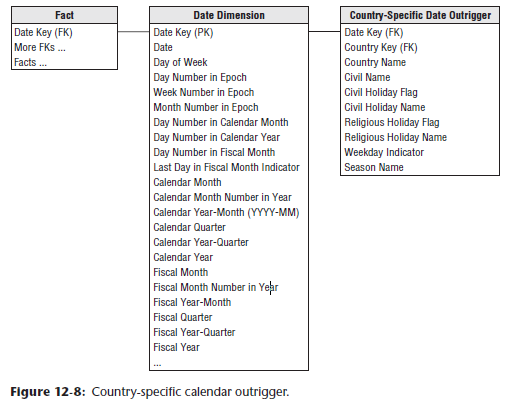
* **2) Combine** the origin + destination airport attributes, + the supplemental city-pair route attributes, **into a single dimension**
* *Theoretically*, the combined dimension could have as many rows as the Cartesian product of all the origin and destination airports
* Fortunately, IRL the number of rows is much smaller than this theoretical limit because airlines don’t operate flights between every airport where they have a presence
* **However, with a couple dozen attributes about the origin airport, + a couple dozen identical attributes about the destination airport, along w/ attributes about the route, you’d probably be more tempted to treat them as separate dimensions**
* Sometimes designers suggest using a **bridge table** containing the origin + destination airport keys to capture route information
* **Although the origin + destination represent a many-to-many relationship, in this case, you can cleanly represent the relationship within the existing fact table rather than using a bridge**

### More Date and Time Considerations

* As often discussed, it’s **important to have a verbose date dimension, whether at the individual day, week, or month granularity, that contains descriptive attributes about the date and private labels for fiscal periods and work holidays**
* There’re also **several *additional* considerations for dealing with date and time dimensions**

#### Country-Specific Calendars as Outriggers

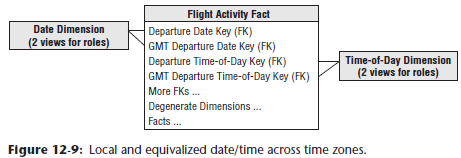
* If the DW/BI system serves **multinational needs**, you **must generalize the standard date dimension to handle multinational calendars in an *open-ended number* of countries**.
* The ***primary* date dimension contains *generic* calendar attributes about the date, *regardless of the country***
* If your multinational business spans Gregorian, Hebrew, Islamic, + Chinese calendars, you’d include 4 sets of days, months, + years in this primary dimension
* **Country-specific date dimensions *supplement* the primary date** **table**
* **The key to the supplemental dimensions = the primary date key, along w/ the country code**
* The table would **include country-specific date attributes**, such as holiday or season names, as illustrated below



* This approach is **similar to the handling of multiple fiscal accounting** **calendars**, as described in Chapter 7: Accounting.
* You **can JOIN this table to the *main* calendar dimension as an outrigger *or* *directly* to the fact table**
* If you provide an interface that requires the user to specify a country name, then the attributes of the country-specific supplement can be viewed as logically appended to the primary date table, allowing them to view the calendar through the eyes of a single country at a time
* **Country-specific calendars can be messy to build in their own right, + things get even more complicated if you need to deal with local holidays that occur on different days in different parts of a country**

#### Date and Time in Multiple Time Zones

* When operating in multiple countries or even just multiple *time zones*, you’re faced with **a quandary concerning transaction dates and times**
* *Do you capture dates + times relative to local midnight in each time zone, or do you express the time period relative to a standard, such as the corporate HQ date/time Greenwich Mean Time (GMT), or Coordinated Universal Time (UTC), also known as Zulu time in the aviation world?*
* **To fully satisfy users’ requirements, the correct answer is probably both**
* **Standard time enables you to see the simultaneous nature of transactions across the business**, whereas **local time enables you to understand transaction timing relative to the time of day**
* Contrary to popular belief, there are more than 24 time zones in the world
* Ex: There is a single time zone in China despite its latitudinal span, there is a single time zone in India, off-set from UTC by 5.5 hours, Australia has 3 time zones, with its Central time zone off set by one-half hour, + meanwhile Nepal + some other nations use one-quarter hour off set
* **Situation gets even more unpleasant when accounting for switches to + from daylight saving time**
* Given the complexities, it’s **unreasonable to think that merely providing a UTC offset in a fact table can support equivalized dates and times**
* Likewise, the **offset can’t reside in a time or airport dimension table because the offset depends on *both* location and date**
* **Recommended approach for expressing dates and times in multiple time zones = include separate date and time-of-day dimensions corresponding to the local and equivalized dates, as below**:



* The time-of-day dimensions, as discussed in Chapter 3: Retail Sales, support time period groupings such as shift numbers or rush period time block designations.

### Location Recap

* We’ve discussed the challenges of international DW/BI system in several chapters of the book
* In addition to the international time zones + calendars discussed in the previous 2 sections, there’s also multi-currency reporting in Chapter 6 and multi-language support in Chapter 8: CRM
* **All these database-centric techniques fall under the general theme of localization**
* **Localization in the larger sense *also* includes the translation of UI text embedded in BI tools**
* BI tool vendors implement this form of localization with text databases containing all the text prompts + labels needed by the tool, which can then be configured for each local environment
* Of course, this can become quite complicated because text translated from English to most European languages results in text strings that are longer than their English equivalents, which may force a redesign of the BI application
* Also, Arabic text reads from right to left, + many Asian languages are completely different.
* **A serious international DW/BI system built to serve business users in many countries needs to be thoughtfully designed to account for a selected set of these localization issues**
* But **perhaps it is worth thinking about how airport control towers + airplane pilots around the world deal w/ language incompatibilities when communicating critical messages about flight directions and altitudes 🡪 *They all use one language (English) and unit of measure (feet)***