CIS 9440 - Data Warehousing and Analytics

Class #5

Week 5 Class Overview:

1. Technical Interview Practice #1
2. SQL Rank Statement
3. Dimensional Modeling - Dimensions
4. Dimensional Modeling - Schemas

Week 5 Class Overview:

1. **Technical Interview Practice #1**
2. SQL Rank Statement
3. Dimensional Modeling - Dimensions
4. Dimensional Modeling - Schemas

What is a Technical Interview?

- For jobs with technical job responsibilities, oftentimes a Technical Interview is conducted after the Behavioral Interview
- Order of Interviews:
  - Screening
  - Behavioral
  - Technical
  - Manager
  - Team

What is a Technical Interview?

- What may a Technical Interview look like?
  - In-person coding on a whiteboard
  - Virtual coding without text editor/compiler
  - Coding test with software/compiler
  - Take home technical assignment

What topics will be tested during a Technical Interview?

1. Specific Domain knowledge
2. Statistics
3. SQL
4. Programming language

How to prepare for a Technical Interview?

- First and foremost, practice the skills you will be tested on. If you will be tested on SQL, practice writing queries!
- Second, you may buy Technical Interview books. This is optional.
  - “Cracking the Coding Interview”
- Third, you may practice with Technical Interview online sites
  - Pramp
  - Interviewing.io

Today’s Technical Interview Practice

Question #1, together - https://www.testdome.com/questions/sql/students/68927?visibility=3&skillId=1483&orderBy=Difficulty

Today’s Technical Interview Practice

Question #2, individual - https://www.testdome.com/questions/sql/web-shop/69934?visibility=3&skillId=1483

Week 5 Class Overview:

1. Technical Interview Practice #1
2. **SQL Rank Statement**
3. Dimensional Modeling - Dimensions
4. Dimensional Modeling - Schemas

What is a RANK statement?

Rank allows you to create a column that ranks another column from largest to smallest (or vice versa). The largest would be ranked 1 and the smallest would be the last number.

What is a RANK statement?

Rank also allows you Partition your rankings. This allows you to rank within subgroups in your data.

Example: Rank animals from largest to smallest. Then partition your rankings by species.

RANK

```
SELECT gameId,
homeTeamName,
awayTeamName,
duration_minutes,
RANK() OVER (ORDER BY duration_minutes DESC) duration_rank
FROM `bigquery-public-data.baseball.schedules`
ORDER BY duration_rank ASC;
```

RANK (example 2)

```
SELECT gameId,
homeTeamName,
awayTeamName,
duration_minutes,
RANK() OVER (ORDER BY duration_minutes DESC) duration_rank,
RANK() OVER (PARTITION BY homeTeamName
  ORDER BY duration_minutes DESC) home_team_duration_rank
FROM `bigquery-public-data.baseball.schedules`
ORDER BY duration_rank ASC;
```

Practice Question:

(5 minutes)

Using the schedules table from the baseball dataset, write a query that returns the `homeTeamName`, `awayTeamName`, and ranks the `attendance` of only games played at night.

Final Project Milestone #1 Feedback:

1. Great work! Impressed.
2. Narrow your scope for more depth
3. KPI's should have value and context

Week 5 Class Overview:

1. Technical Interview Practice #1
2. SQL Rank Statement
3. **Dimensional Modeling - Dimensions**
4. Dimensional Modeling - Schemas

Dimensions - Definition and Examples

Dimension: Dimensions are entities that establish context for measures. Unlike measures, which are numeric, dimensions are descriptive. Think about Who, What, Where, When, and How.

Common Dimensions:

- Date
- Time
- Geography (Location)
- Product
- Employees
- Customers

Dimensions - Definition and Examples

Example Fact: Customer Support Inquiry

- Each row is one inquiry from a customer of your organization's business

What Dimensions could we Join to this Fact table to drive insights?

Dimensions - Hierarchies

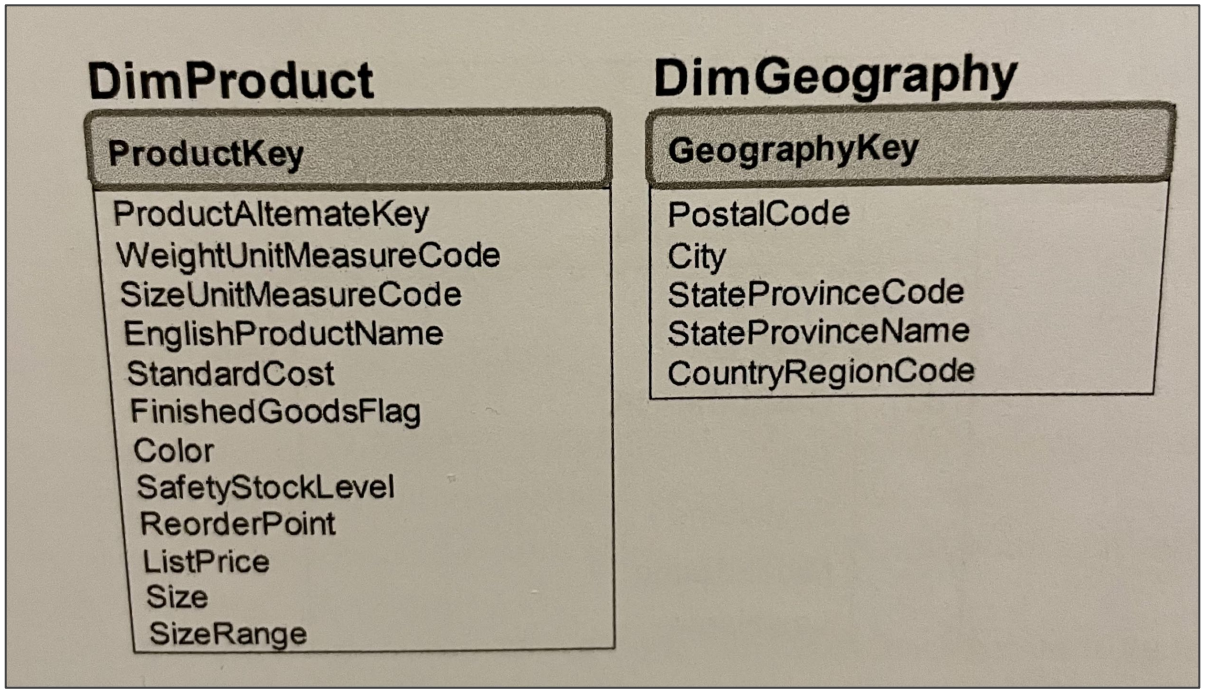
Dimensions are oftentimes hierarchical; they group things together in ways that an enterprise would measure itself

Dimensions - Hierarchies example

Revisit the NYC Library example, here may be the Location Dimension Table, with a hierarchy from zip code to state

DimLocation
LocationKey **(PK)**
Location_Name
Zip_Code
Neighborhood
Borough
City
State

Dimensions - How are they linked?



Each dimension is connected to the fact table through their primary key (PK to FK relationship)

When you join the Sales fact table to the Product table, now you can view sales by any of the attributes in the dimension

Dimensions - Keys and Surrogate Keys

To link each Dimension to the Fact Table, you must use a Key

- Should your key be meaningful, or meaningless?
  - For example, could you use a purchase order ID as a Primary Key? Or a Product ID?

Dimensions - Keys and Surrogate Keys

- Best practice is to use made up, meaningless keys called **surrogate keys**
    - Exceptions! The date dimension and time dimension tables may use meaningful keys

Surrogate Keys

- What could go wrong with Zip Code?
  - What if one source system does not have Zip Code?
  - What if a Zip Code is changed?
  - What if information from a Zip Code is no longer in the source system?

DimLocation
LocationKey **(PK)**
Zip_Code
Borough
City
State
Country

Surrogate Keys

- Why can you use meaningful keys (non-surrogate keys) for Date and Time dimensions?

date_dim
date_id **(PK)** **(YYYYMMDD)**
date
day_of_week
is_weekend
week_number
month
quarter
year
decade

time_dim
time_id **(PK)** **(HHMM)**
minute
hour
is_morning
is_working_hours

10 Minute Break

Dimensional Modeling – Slowly Changing Dimensions

- Some Dimensions need special treatment: Specifically those that have attributes that change over time
- E.g. Products change price, cost
- E.g., Employees are promoted, fired, change departments, get new titles
- E.g., Customers change addresses, names, get married, move into different income brackets
- Such changes don't happen that often; they occur infrequently and unpredictably

Slowly Changing Dimensions: How to Handle?

- There are 7 ways to handle Slowly Changing Dimensions, labelled at 7 SCDs
- Some of easier to implement than others
  - But each have tradeoffs
    - Example, how much historical context do you wish to keep?

Slowly Changing Dimension Type 0 - Fixed Dimension

- Type 0: Do Nothing & Keep Original! Even if a change occurs, do nothing; we simply never change the dimensional data after the initial load.
    - Example: The Date Dimension

Slowly Changing Dimension Type 1 - No History

- Type 1: Overwrite (or Update in Place)
  - No history of OLTP data in the DW; simply take new OLTP data and overwrite what's in DW
  - Type 1 changes are appropriate for correcting errors and for situations where a conscious choice is made not to track history. And of course, most data warehouses start out with Type 1 as the default.

Slowly Changing Dimension
Type 1 - No History (Example)

Initial Row

Employee Key	First Name	Last Name	Title	HireDate	Marital Status	Base Rate	Row Created	Row Modified
12345	Guy	Gilbert	Prod Technician	08/01/07	S	$12.45	08/01/07	

Current Row

Employee Key	First Name	Last Name	Title	HireDate	Marital Status	Base Rate	Row Created	Row Modified
12345	Guy	Gilbert	Prod Supervisor	08/01/07	S	$25.00	08/01/07	01/02/14

- Not keeping history; Just an overwrite! So we have no idea when the employee was promoted, when he was married or divorced, or when his base rate pay increased. In fact, his salary could easily have gone up multiple times, maybe even annually, but with this technique we don't see any of that information.
- Easy to implement & does not create additional dimension rows;
- HOWEVER, any aggregate fact tables or OLAP cubes must be recomputed post any updates

Slowly Changing Dimension Type 2 - Row Versioning

- Creates a new row in the dimension table every time a change occurs in an attribute that we wish to track.
- Unlike type 1, it enables you to track history accurately, so you see every time an attribute changes in that row and for that employee
- This technique keeps track of all the historical changes in the dimension, allowing you to use the values of the dimension attributes at any point in time and examine the impact of those changes on the relevant fact tables. Facilitates ‘as-is’ & ‘as-was’ analysis
- Need to use (In addition to a surrogate key) additional administrative fields to assist:
- Begin Effective Date
- End Effective Date
- Current Flag
- Reason for Change

Slowly Changing Dimension
Type 2 - Row Versioning (example)

Last Name	Title	Hire Date	Marital Status	Base Rate	Effective Date	Ineffective Date	Status
Gilbert	Prod Technician	08/01/07	S	$12.45	08/01/07		Current

After Life & Time Happens…

Last Name	Title	Hire Date	Marital Status	Base Rate	Effective Date	Ineffective Date	Status
Gilbert	Prod Technician	08/01/07	S	$12.45	08/01/07	06/11/08	Expired
Gilbert	Prod Technician	08/01/07	M	$12.45	06/12/08	01/31/10	Expired
Gilbert	Prod Technician	08/01/07	M	$17.50	02/01/10	01/01/14	Expired
Gilbert	Prod Supervisor	08/01/07	S	$25.00	01/02/14		Current

Slowly Changing Dimension Type 2 - Row Versioning (example continued)

Dimension Table SK	Dimension Durable SK	Dimension NK	SOR SK	Effective Date	Ineffective Date	Current Indicator	Dimension Attributes
20123	12345	ABC123	2	1/2/2007	2/3/2009	N	
21357	12345	ABC123	2	2/4/2009	6/22/2012	N	
23488	12345	ABC123	2	6/23/2012	1/1/2014	N	
24719	12345	ABC123	2	1/2/2014	12/31/9999	Y	

f10-16

How Dimension would be stored

Fact Table

Column	Key
Dimension_Table_SK	FK
Dimension_Durable_SK	FK
Other Dimension's FKs	FK
Other Attributes...	

Dimension - Type 2

Column	Key
Dimension_Table_SK	PK
Dimension_Durable_SK	FK
Dimension_NK	NK
SOR_SK	FK
Effective Date	
Ineffective Date	
Current Indicator	
Other Dimension Attributes	

f10-17

How Fact Table & Dimension would be linked

- **To get the values at the point of time that the fact occurred "as was,"** select the row based on *Dimension_Table_SK*
- **To get the current values,** select the row based on *Dimension_Durable_SK* and where the Current_Indicator = "Y" (or whatever flag was used to indicate it is the current row)
- **To get the value at any particular point in time,** select the row based on *Dimension_Table_SK* with the date of interest being between the *effective date* and *ineffective date*

Slowly Changing Dimension
Type 3 - Previous Value Column

Initial Row

Employee Key	First Name	Last Name	Title	Original Title	HireDate	Marital Status	Base Rate	Effective Date	Ineffective Date
12345	Guy	Gilbert	Prod Technician	Prod Technician	08/01/07	S	$12.45	08/01/07	12/31/99

Current Row

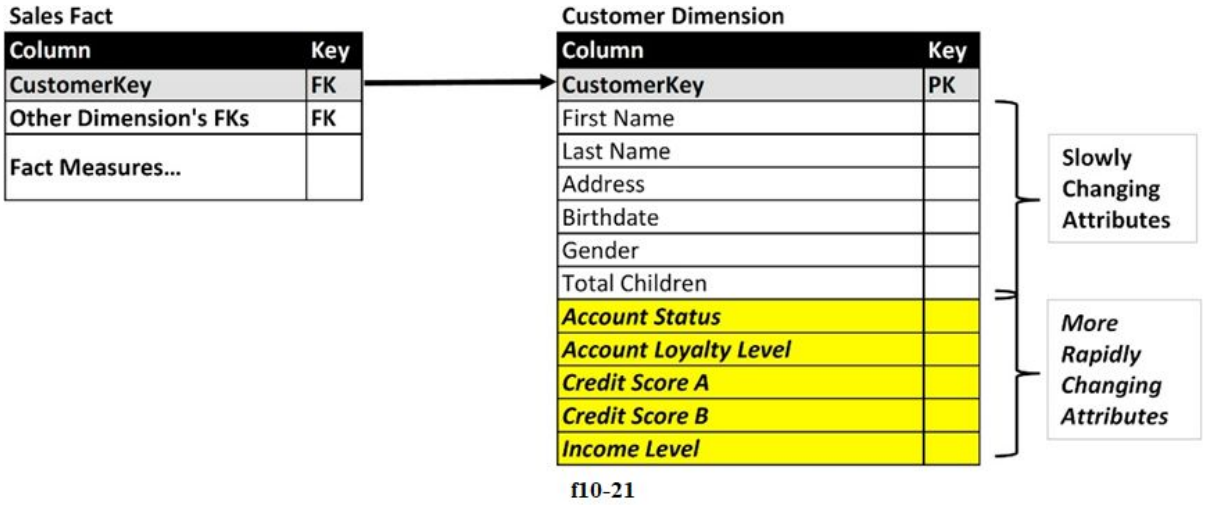
Employee Key	First Name	Last Name	Title	Original Title	HireDate	Marital Status	Base Rate	Effective Date	Ineffective Date
12345	Guy	Gilbert	Prod Supervisor	Prod Technician	08/01/07	S	$25.00	01/02/14	12/31/99

Note that we are tracking the title changes (but only to one degree)

Note that the base rate changed also but we have no history!

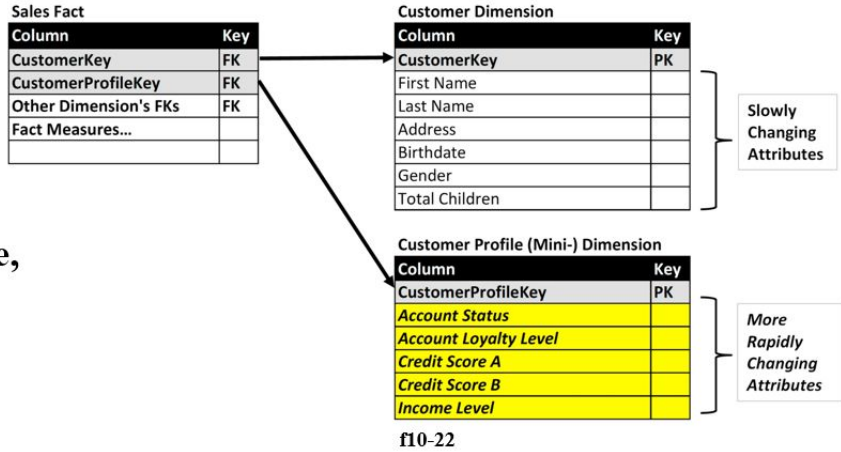
Slowly Changing Dimension Type 4 - History Table

- Type 4: Keep a Historical Table for some Dimension Attributes that change more often than others



Slowly Changing Dimension Type 4 - History Table

- Approach is to group the fast-changing dimension attributes into a separate mini-dimension
- Then treat that mini-dimension as a “type 2” style dimension
- The fact table would link to both the main Dimension AND the mini-dimension by way of FKs



Note that in this case, you cannot get from Dimension to Mini-Dim or vice-versa!

f10-22

Slowly Changing Dimension Summary

SCD Type	Dimension Table Action	Impact on Fact Analysis
Type 0	No change to attribute value	Facts associated with attribute's original value
Type 1	Overwrite attribute value	Facts associated with attribute's current value
Type 2	Add new dimension row for profile with new attribute value	Facts associated with attribute value in effect when fact occurred
Type 3	Add new column to preserve attribute's current and prior values	Facts associated with both current and prior attribute alternative values
Type 4	Add mini-dimension table containing rapidly changing attributes	Facts associated with rapidly changing attributes in effect when fact occurred
Type 5	Add type 4 mini-dimension, along with overwritten type 1 mini-dimension key in base dimension	Facts associated with rapidly changing attributes in effect when fact occurred, plus current rapidly changing attribute values
Type 6	Add type 1 overwritten attributes to type 2 dimension row, and overwrite all prior dimension rows	Facts associated with attribute value in effect when fact occurred, plus current values
Type 7	Add type 2 dimension row with new attribute value, plus view limited to current rows and/or attribute values	Facts associated with attribute value in effect when fact occurred, plus current values

Creating a Date Dimension in BigQuery

```sql
-- create date_dim in bigquery
SELECT
  CONCAT (FORMAT_DATE("%Y",d),FORMAT_DATE("%m",d),FORMAT_DATE("%d",d)) as date_id,
  d AS full_date,
  FORMAT_DATE('%w', d) AS week_day,
  FORMAT_DATE('%A', d) AS day_name,
  EXTRACT(DAY FROM d) AS year_day,
  EXTRACT(WEEK FROM d) AS week,
  EXTRACT(WEEK FROM d) AS year_week,
  EXTRACT(MONTH FROM d) AS month,
  FORMAT_DATE('%B', d) as month_name,
  FORMAT_DATE('%Q', d) as fiscal_qtr,
  EXTRACT(YEAR FROM d) AS year,
  CONCAT (EXTRACT(YEAR FROM d), EXTRACT(WEEK FROM d)) AS year_week_concat,
  CONCAT (EXTRACT(YEAR FROM d), EXTRACT(MONTH FROM d)) AS year_month_concat,
  (CASE WHEN FORMAT_DATE('%A', d) IN ('Sunday', 'Saturday') THEN 0 ELSE 1 END) AS day_is_weekday,
FROM (
  SELECT
    *
  FROM
    UNNEST(GENERATE_DATE_ARRAY('2014-01-01', '2030-01-01', INTERVAL 1 DAY)) AS d )
```

Week 5 Class Overview:

1. Technical Interview Practice #1
2. SQL Rank Statement
3. Dimensional Modeling - Dimensions
4. **Dimensional Modeling - Schemas**

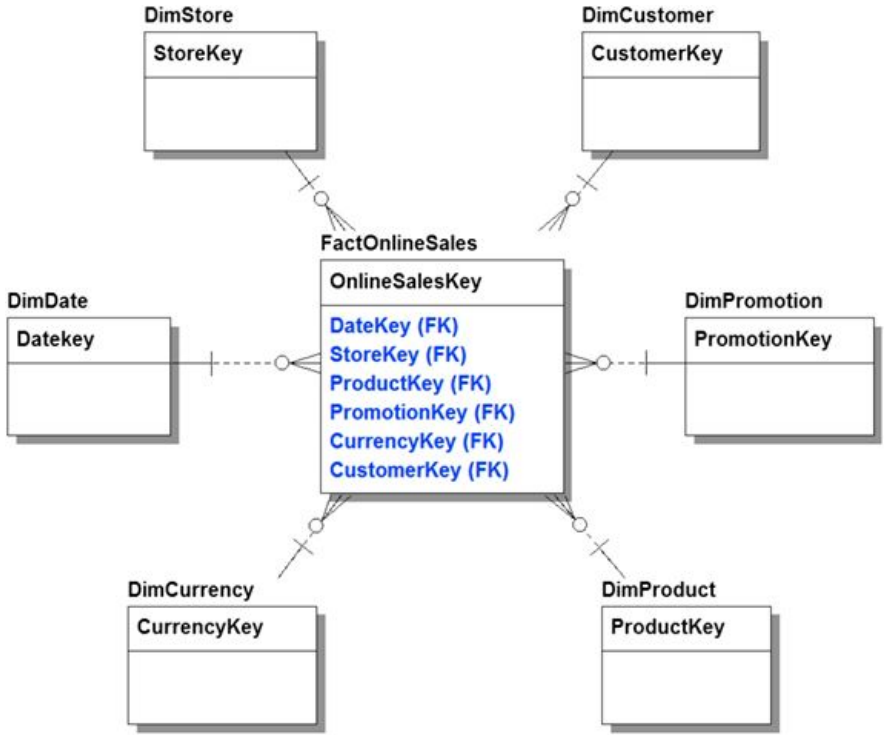
Dimensional Modeling - Schemas

- You have seen Fact Tables, Dimensions, and Attributes. Now we will put them together with Schemas!

- **Schema:** the organizational structure of tables in a database.
  - How do you connect your Facts and Dimensions?

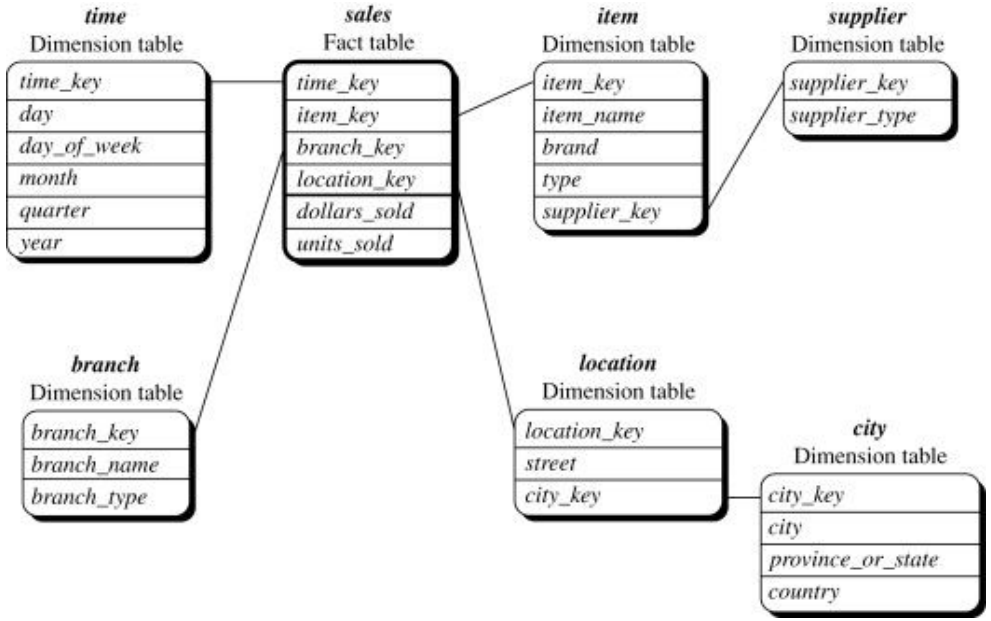
Schemas - Star Schema

- The most common type of schema
- Fact surrounded by Dimensions
- A compromise between normalized and denormalized
    - Fact tables are normalized, dimensions are not
- Advantages: performance, ease to query, analytics, **intuitive** (similar to a pivot table)



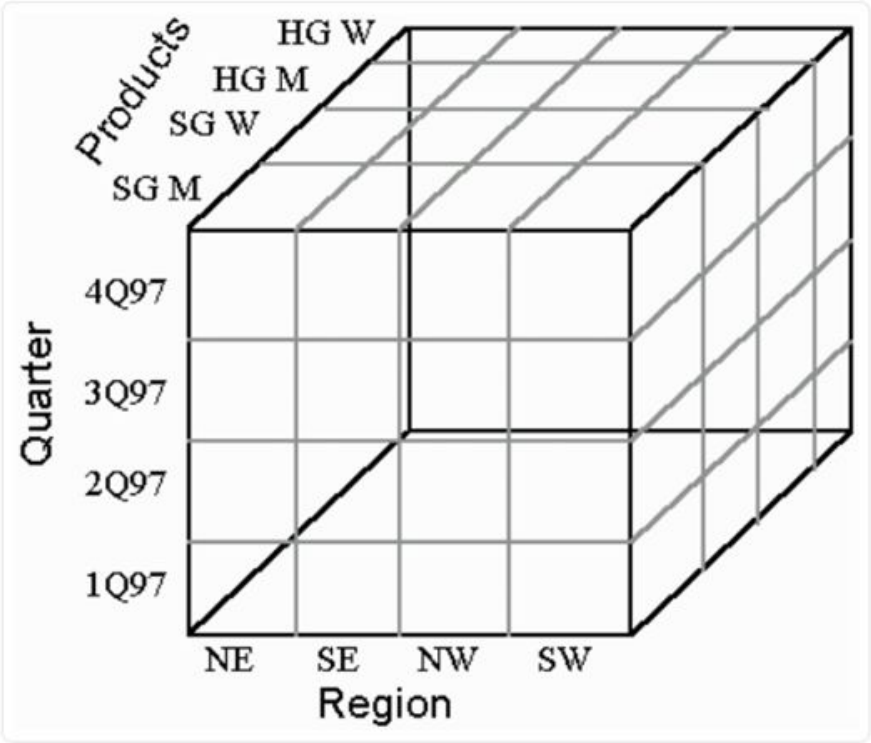
Schemas - Snowflake Schema

- Fact table surrounded by **normalized** Dimensions
  - Break down the hierarchies
- Disadvantages: Less intuitive, can get much larger
- Advantages: optimized for certain BI tools, which perform very well with snowflake models



Schemas - Multidimensional Schema

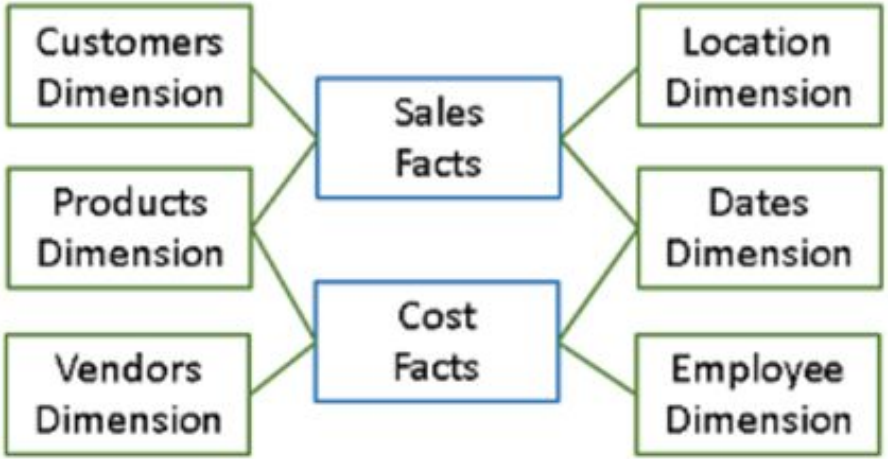
- Often called OLAP Cubes
- Think about the data stored like a pivot table
- These are uniquely stored in multidimensional databases, unlike star and snowflake in relational databases
- They’re proprietary, custom built
- Good for pre-calculating big data that’s queried often (MSTR)



Cube showing 3 dimensions: Products, Quarter 1997 and Region

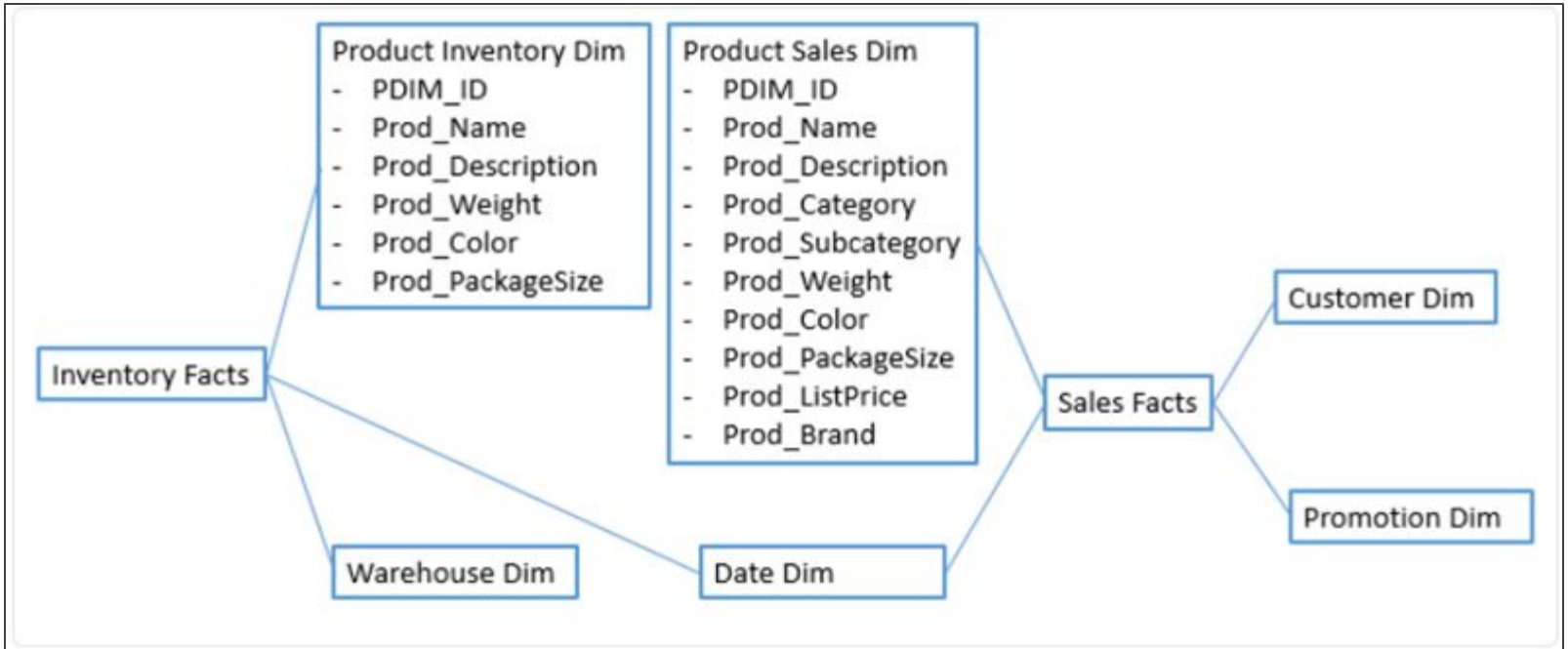
Schemas - Multifact Star Models

- In reality, you will have more than one fact table
- Thus, you will share dimensions among multiple fact tables - called **conformed dimensions**
- This is supported by **BUS matrix** in Kimball Lifecycle

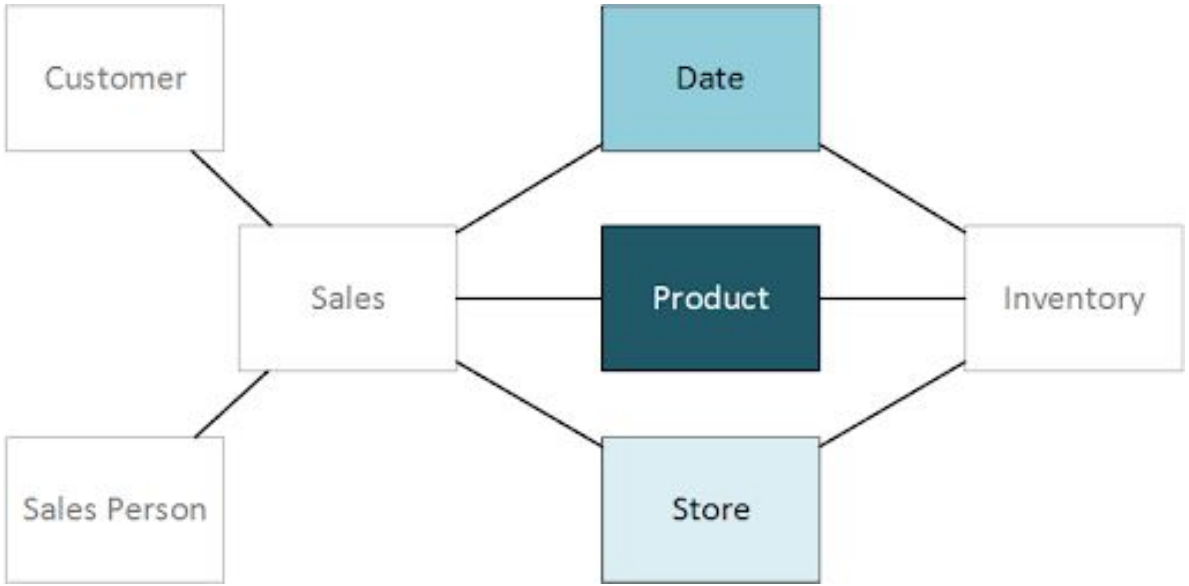


What is a Conformed Dimension?

- **Conformed Dimension:** a dimension used by multiple Fact Tables
  - Below: One product dimension must be created for best reporting - conformed dimension



What is a Conformed Dimension? Example 2



Kimball BUS Matrix

BUS Matrix for Data Warehouse

BUSINESS PROCESSES	COMMON DIMENSIONS						
	Date	Product	Warehouse	Store	Promotion	Customer	Employee
Issue Purchase Orders	X	X	X				
Receive Warehouse Deliveries	X	X	X				X
Warehouse Inventory	X	X	X				
Receive Store Deliveries	X	X	X	X			X
Store Inventory	X	X		X			
Retail Sales	X	X		X	X	X	X
Retail Sales Forecast	X	X		X			
Retail Promotion Tracking	X	X		X	X		
Customer Returns	X	X		X	X	X	X
Returns to Vendor	X	X		X			X
Frequent Shopper Sign-Ups	X			X		X	X

- Where are the facts and dimensions?
- How to start?
- When to add?
- What if facts are not all at same hierarchy level?

BUS Matrix for Final Project

	Common Dimensions				
Business Processes (Facts)	Date	Product	Store	Customer	Employee
Sales	X	X	X	X	X
Inventory	X	X	X		
Costs	X	X			X

Link: https://docs.google.com/spreadsheets/d/1u0qVE4Th_HZ8ES4aEd0n0kGMJNNE8Nd49oPEGNRmGA4/edit?usp=sharing

Link: https://docs.google.com/spreadsheets/d/1u0qVE4Th_HZ8ES4aEd0n0kGMJNNE8Nd49oPEGNRmGA4/edit?usp=sharing

Create Kimball BUS Matrix

DW Goal: Draw insights about NYC Library Books

- Fact Tables:
  - Books Checked Out
  - Library Book Inventory
  - Library Book Orders

- Dimensions:
  - Time
  - Date
  - Author
  - Genre
  - Location
  - Members
  - Employees

Creating a Dimensional Model

Goal: Dimensional Model data about Albums from Spotify

- Data was captured from Spotify's API with python, a typical way data may enter your Data Warehouse from an outside source

label	name	popularity	total_tracks	artist	album_release_date
EMI Catalogue	Please Please Me (Remastered)	77	14	The Beatles	19630322
EMI Catalogue	With The Beatles (Remastered)	72	14	The Beatles	19631122
ABKCO (US)	The Rolling Stones	50	12	The Rolling Stones	19640416
EMI Catalogue	A Hard Day's Night (Remastered)	74	13	The Beatles	19640710
ABKCO Music and Records, Inc.	12 X 5	50	12	The Rolling Stones	19641017

https://docs.google.com/spreadsheets/d/1H-H8nuJPbaSENpAAHMKpO7hJVyDZwhByJMQIn9WBIXA/edit?usp=sharing

First Step: Define your KPI's

- Hypothetically, we work for the a Historical Music Organization.
- We're creating a Data Warehouse to measure the most popular Rock music of All Time.
- Let's define some KPI's to measure music popularity
    - Example, Total Albums by Artist

label	name	popularity	total_tracks	artist	album_release_date
EMI Catalogue	Please Please Me (Remastered)	77	14	The Beatles	19630322
EMI Catalogue	With The Beatles (Remastered)	72	14	The Beatles	19631122
ABKCO (US)	The Rolling Stones	50	12	The Rolling Stones	19640416
EMI Catalogue	A Hard Day's Night (Remastered)	74	13	The Beatles	19640710
ABKCO Music and Records, Inc.	12 X 5	50	12	The Rolling Stones	19641017

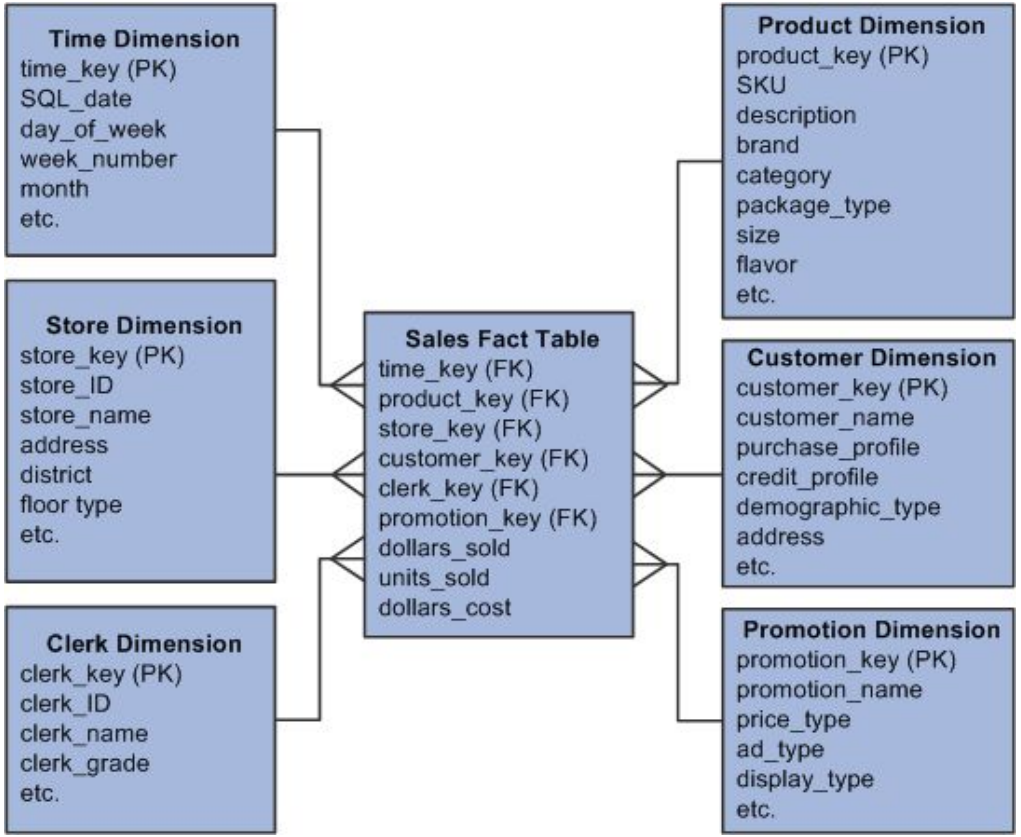
KPI’s

1. TBD
2. TBD
3. TBD

(KPI’s in public chat)

label	name	popularity	total_tracks	artist	album_release_date
EMI Catalogue	Please Please Me (Remastered)	77	14	The Beatles	19630322
EMI Catalogue	With The Beatles (Remastered)	72	14	The Beatles	19631122
ABKCO (US)	The Rolling Stones	50	12	The Rolling Stones	19640416
EMI Catalogue	A Hard Day's Night (Remastered)	74	13	The Beatles	19640710
ABKCO Music and Records, Inc.	12 X 5	50	12	The Rolling Stones	19641017

Time to make a Star Schema (1 Fact)



https://docs.google.com/drawings/d/1eW6Em1cVtgVJDxzvNftC5hq9_ugSidwhNFZRBTtnLtc/edit?usp=sharing

Remember your Keys and to figure out Facts and Dimensions by dissecting each KPI

Homework:

1. Final Project Milestone #2 on Blackboard, due 10/22/22