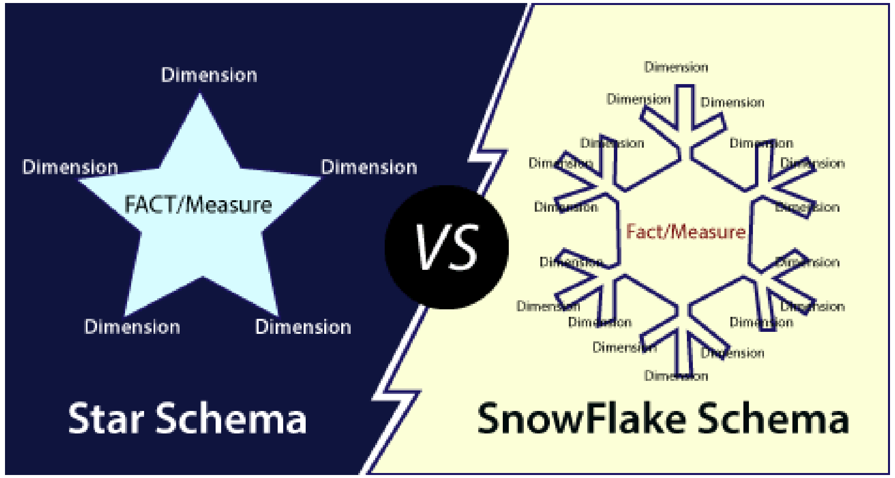
BIG DATA AND ANALYTICS

DATA WAREHOUSE SCHEMAS FOR DECISION MAKING: STAR AND SNOWFLAKE SCHEMA

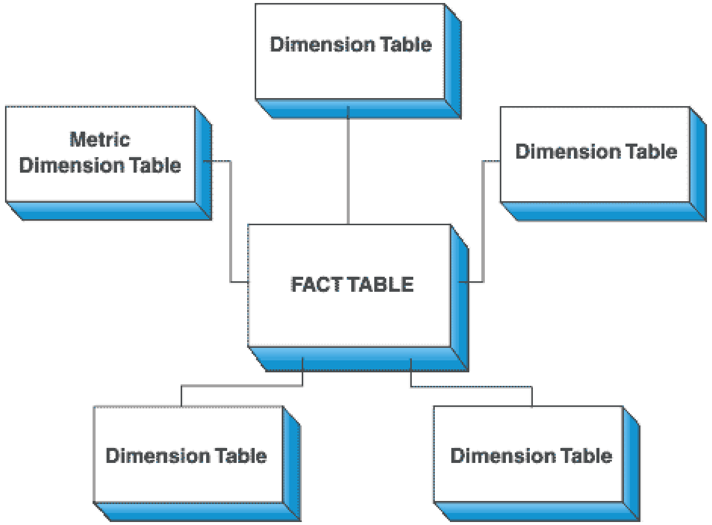
Definitions

- When it comes to snowflake schemas vs. star schemas, it's essential to remember their basic definitions:
  - **star schemas:** a schema in data warehouse, in which the center of the star can have one fact table and a number of associated dimension tables.
    - Offer an efficient way to organize information in a data warehouse
  - **snowflake schemas:** a schema in data warehouse in which a logical arrangement of tables in a multidimensional database such that the ER diagram resembles a snowflake shape.
    - Are a variation of star schemas that allow for more efficient data processing.
  - Both schemas improve the speed and simplicity of read queries and complex data analysis-especially when dealing with large data sets that pull information from diverse sources.
  - Despite their similarities, star and snowflake schemas have key differences we all need to understand.



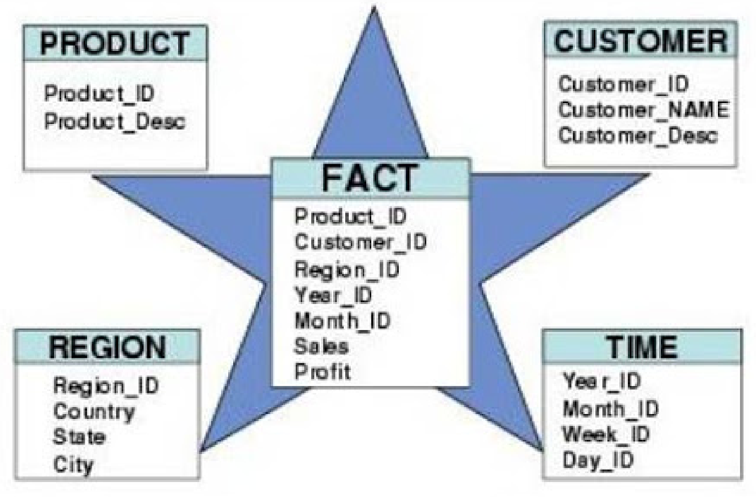
Star Schema

- Star schemas offer the simplest structure for organizing data into a data warehouse.
- It is known as star schema as its structure resembles a star.
- The center of a star schema consists of one or multiple **fact tables** that contains measurable or countable fact data that index a series of **dimension tables**.
- The fact data gets organized into fact tables, and the dimensional data into dimension tables.
- The purpose of a star schema is to cull out numerical ***fact data*** relating to a business and separate it from the descriptive, or ***dimensional data***.
  - **Fact data** will include information like price, weight, speed, and quantities - i.e., data in a numerical format.
  - **Dimensional data** will include uncountable things like colours, model names, geographical locations, employee names, salesperson names, etc., along with the numerical information



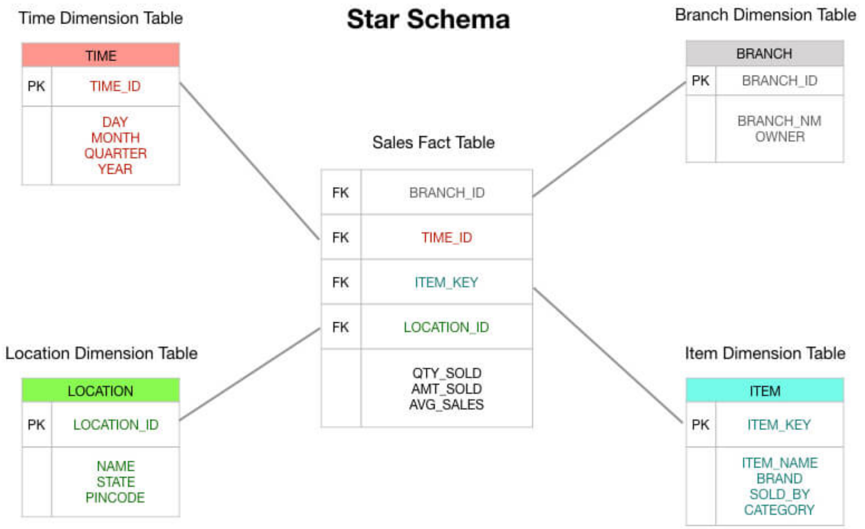
Star Schema …

- The fact table contains foreign keys to join to dimension data in a metric dimension table and in one or more component dimension tables
- Fact tables are the points of integration at the center of the star schema in the data warehouse.
  - They allow machine learning tools to analyze the data as a single unit, and they allow other business systems to access the data together.
- Dimension tables hold and manage the data (numerical and nonnumerical) which converges through fact tables that make up the data warehouse.
- Fact tables keep track of numerical information related to different events.
  - E.g. they might include numeric values along with foreign keys that map to additional (descriptive and nonnumerical) information in the dimension tables.
- Fact tables maintain a low level of granularity (or detail), which is to say, they record information at a more atomic level.
  - This could lead to the build-up of many records within the fact table over time



Types of Fact Tables

- There are three main kinds of fact tables:
  - **Transaction fact tables**: are easy to understand: a customer or business process does some thing; you want to capture the occurrence of that thing, and so you record a transaction in your data warehouse and you're good to go.
    - These receive a transaction, you record the transaction in your fact table, and this becomes the basis of your reporting.
  - **Snapshot fact tables**: These record information that applies to specific moments in time(captures some sort of periodic data), like year-end account statements, daily snapshot of financial metrics, or perhaps a weekly summary of accounts receivable.
    - Note that if no transactions occur during a certain period, a new row *must* be inserted into the periodic snapshot table, even if every fact that is saved is a null!
  - **Accumulating snapshot tables**: These record information related to a running tally of data, like year-to-date sales figures for specific merchandise or categories of merchandise.



Star Schema

Time Dimension Table

	TIME
PK	TIME_ID
	DAY MONTH QUARTER YEAR

Branch Dimension Table

	BRANCH
PK	BRANCH_ID
	BRANCH_NM OWNER

Sales Fact Table

FK	BRANCH_ID
FK	TIME_ID
FK	ITEM_KEY
FK	LOCATION_ID
	QTY_SOLD AMT_SOLD AVG_SALES

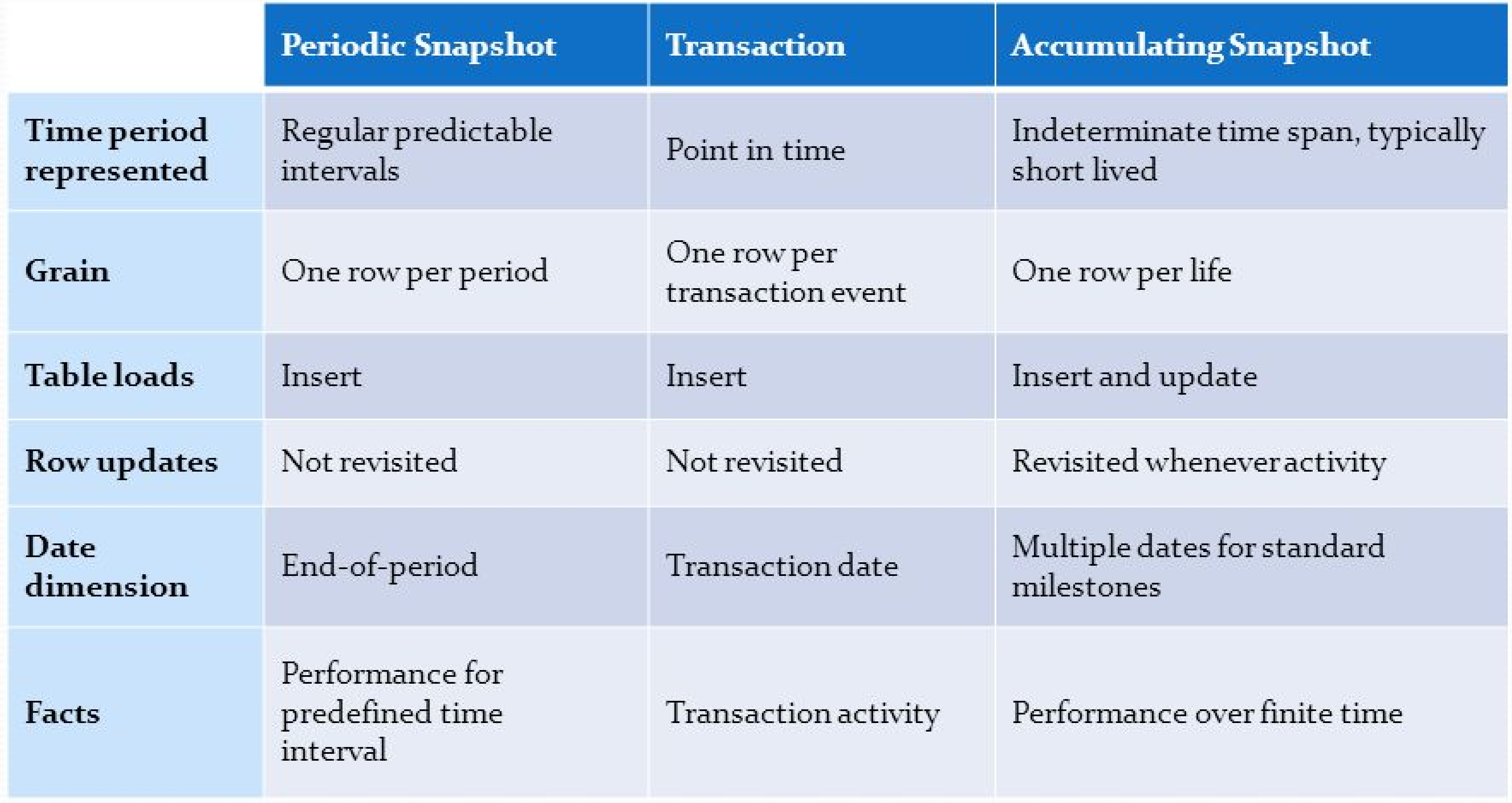
Location Dimension Table

	LOCATION
PK	LOCATION_ID
	NAME STATE PINCODE

Item Dimension Table

	ITEM
PK	ITEM_KEY
	ITEM_NAME BRAND SOLD_BY CATEGORY

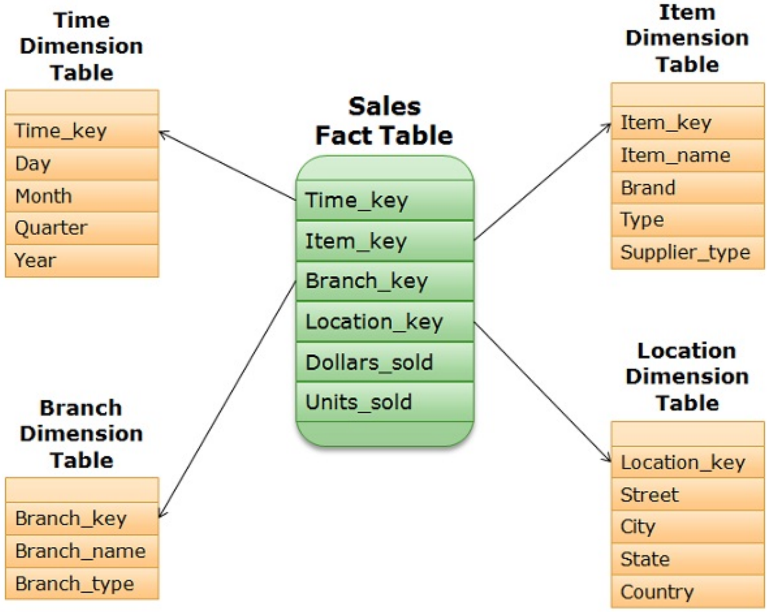
Comparison of Fact Table Types



	Periodic Snapshot	Transaction	Accumulating Snapshot
Time period represented	Regular predictable intervals	Point in time	Indeterminate time span, typically short lived
Grain	One row per period	One row per transaction event	One row per life
Table loads	Insert	Insert	Insert and update
Row updates	Not revisited	Not revisited	Revisited whenever activity
Date dimension	End-of-period	Transaction date	Multiple dates for standard milestones
Facts	Performance for predefined time interval	Transaction activity	Performance over finite time

Types of Dimension Tables

- Dimension tables normally store fewer records than fact tables;
  - records such as descriptive attributes and numerical data
- There are many types of dimension tables depending on the information system. Here are some examples:
  - **Time dimension tables**: Information to identify the exact time, date, month, year different events happened.
  - **Geography dimension tables**: Address/location information.
  - **Employee dimension tables**: Information about employees and salespeople, such as addresses, phone numbers, names, employee numbers, and email addresses.
  - **Merchandise dimension tables**: Descriptive information about products, their product numbers, etc.
  - **Customer dimension tables**: Customer name, numbers, contact information, addresses, etc.
  - **Range dimension tables**: Information relating to a range of values for time, price, and other quantities.



Time Dimension Table
- Time_key
- Day
- Month
- Quarter
- Year

Sales Fact Table
- Time_key
- Item_key
- Branch_key
- Location_key
- Dollars_sold
- Units_sold

Item Dimension Table
- Item_key
- Item_name
- Brand
- Type
- Supplier_type

Branch Dimension Table
- Branch_key
- Branch_name
- Branch_type

Location Dimension Table
- Location_key
- Street
- City
- State
- Country

Characteristics of Star Schema

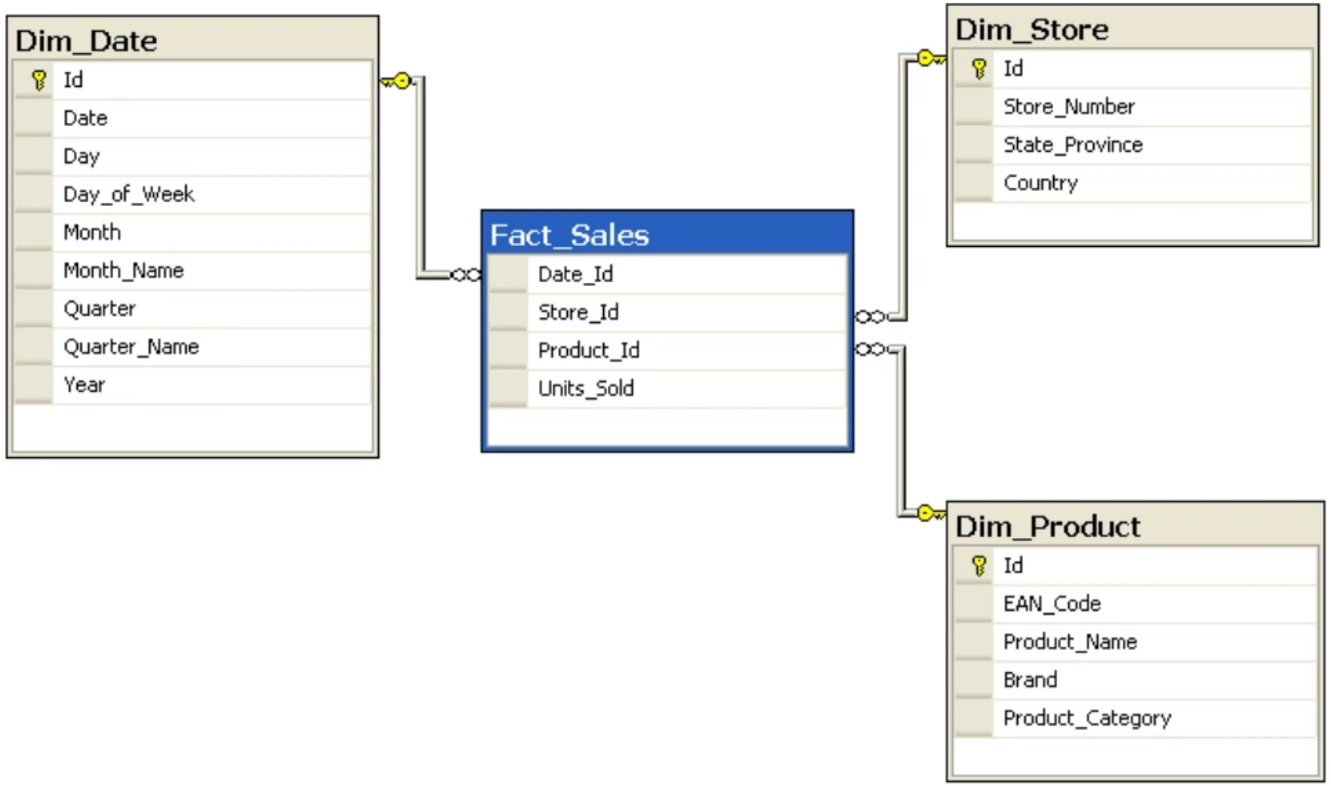
- Every dimension in a star schema is represented with the only one-dimension table.
- The dimension table should contain the set of attributes.
- The dimension table is joined to the fact table using a foreign key
- The dimension table are not joined to each other
- Fact table would contain key and measure
- The Star schema is easy to understand and provides optimal disk usage.
- The dimension tables are not normalized. For instance, in the above figure, Country_ID does not have Country lookup table as an OLTP design would have.
- The schema is widely supported by BI Tools

How Fact Tables and Dimension Tables Work Together

- Dimension tables usually list a surrogate primary key (i.e., a data type that consists of a single-column integer) that maps to attributes related to the natural key.
- Imagine you have a dimension table with information relating to different stores: ***Dim_Store***
- You can assign an ID number to each store and its row of related nonnumerical and other information, like store name, size, location, number of employees, category, etc.
- As it follows, wherever you list the **Store ID** number on the fact table (**Fact_Sales**), it will map to that specific row of store data on the ***Dim_Store*** dimension table.
- Of course, the star schema doesn't stop there, because there are additional points (or dimension tables) with information that links to the fact table.

The following diagram illustrates what a simple star schema looks like

- ✓ Here, the fact table, Fact_Sales, is at the center of the diagram.
- ✓ Its schema includes the following columns for ID numbers: Date_Id, Store_Id, Product_Id, and Units_Sold.
- ✓ As the point of integration, the fact table integrates the diverse information in the dimension tables: Dim_Product, Dim_Store, and Dim_Date.
- ✓ As you can see, the star schema gets its name from having a central fact table “core,” and dimension table “points.”
- ✓ When a star schema has many dimension tables, data engineers might refer to it as a *centipede schema*.



How Fact Tables and Dimension Tables Work Together

- As an example, let's say you want to know the following for the time-period October 2020:
  - How many products were purchased?
  - What products were purchased?
  - In what stores were the products purchased?
  - What were the names and addresses of the products purchased?
  - What brand name manufactured the products purchased?
  - What day of the week did customers make each product purchased?
- To conduct a query like this, you'll need to access data contained in all of the dimension tables (Dim_Date, Dim_Store, and Dim_Product).
- These are separate databases; however, through the fact table (which serves as a point of integration) you can query all of the data, akin to it being in a single table.
- And that's how a star schema data warehouse works!

Denormalization of Data in Star Schemas

- The star schema’s goal is to speed up *read queries* and *analysis* for massive amounts of data contained in diverse databases with different source schemas.
  - The star schema achieves this goal through the “denormalization” of the data within the network of dimension tables.
- Traditionally, database managers sought the “normalization” of data by eliminating duplicate copies of the same data, which is to say, the normalization of the duplicate information into one copy.
  - This made write commands faster because only one copy of the data needed updating.
- When a data system expands into multiple dimension tables, however, accessing and analysing data from multiple sources slows down *read queries* and *analysis.*
  - To speed things up, the star schema relaxes the traditional rules of database normalization by “*de*normalizing” the data.
- A star schema pulls the fact data (or ID number primary keys) from the dimension tables, duplicates this information, and stores it in the fact table.
  - In that way, the fact table connects all of the information sources together.
  - This makes read queries and analysis infinitely faster.
  - However, it sacrifices the speed of *write commands.* The slower write commands happen because the system needs to update all counterpart copies of the “denormalized” data following each update.

Benefits of Star Schemas

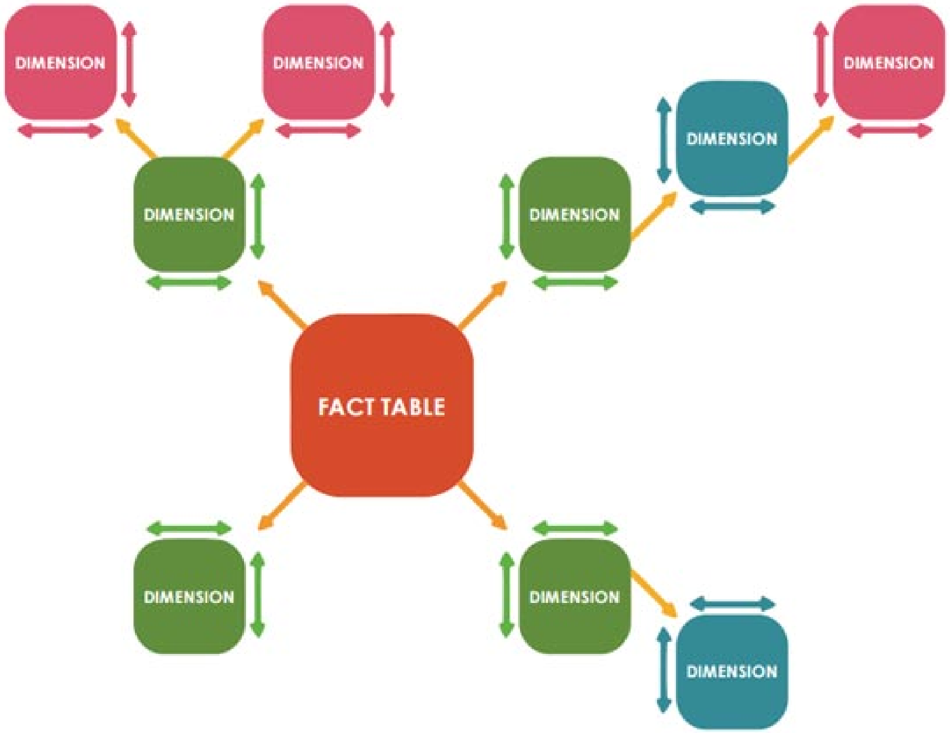
- **Queries are simpler**: Because all of the data connects through the fact table the multiple dimension tables are treated as one large table of information, and that makes queries simpler and easier to perform.
- **Easier business insights reporting**: Star schemas simplify the process of pulling business reports like as-of-as and period-over-period reports.
- **Better-performing queries**: By removing the bottlenecks of a highly normalized schema, query speed increases, and the performance of read-only commands improves.
- **Provides data to OLAP systems**: OLAP (Online Analytical Processing) systems can use star schemas to build OLAP cubes.

Challenges of Star Schemas

- **Decreased data integrity**: Because of the denormalized data structure, star schemas do not enforce data integrity very well.
  - Although star schemas use countermeasures to prevent anomalies from developing, a simple insert or update command can still cause data incongruities.
- **Less capable of handling diverse and complex queries**: Databases designers build and optimize star schemas for specific analytical needs.
  - As denormalized data sets, they work best with a relatively narrow set of simple queries.
  - Comparatively, a normalized schema permits a far wider variety of more complex analytical queries.
- **No Many-to-Many Relationships**: Because they offer a simple dimension schema, star schemas don’t work well for many-to-many data relationships.

Snowflake Schema

- Now that you understand how star schemas work, you're ready to explore the snowflake schema (which takes the shape of a snowflake).
- The purpose of a snowflake schema is to normalize the denormalized data in a star schema.
- This solves the write command slow-downs and other problems typically associated with **star schemas**.
- The snowflake schema is a **multi-dimensional structure**.
  - At its core are fact tables that connect the information found in the dimension tables, which radiate outward like in the star schema.
  - The difference is that the dimension tables in the snowflake schema divide themselves into more than one table. That creates the snowflake pattern.



Snowflake Schema …

- Through this “snowflaking” method, the snowflake schema normalizes the dimension tables it connects with by
  - Firstly, getting rid of low cardinality attributes (that appear multiple times in the parent table); and
  - Then turning the dimension tables into more than one table, until the dimension tables are completely normalized.
- Like snowflake patterns in nature, the snowflake database becomes exceedingly complex.
- The schema can produce elaborate data relationships, where child tables have more than one parent table.

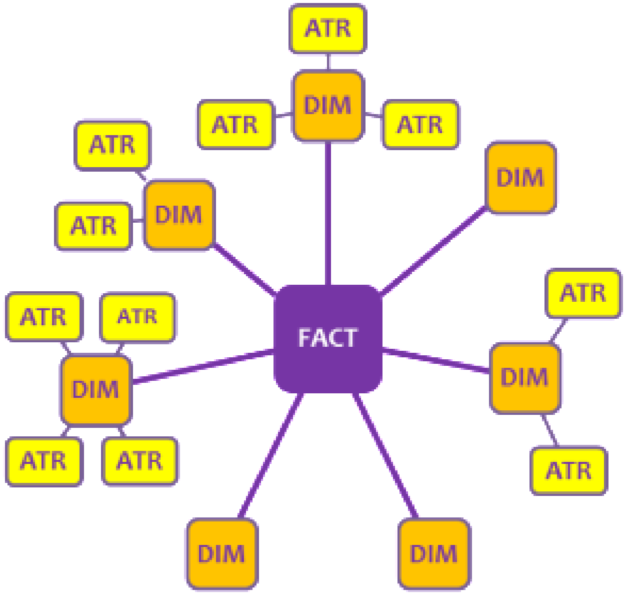
SnowFlake Design Schema

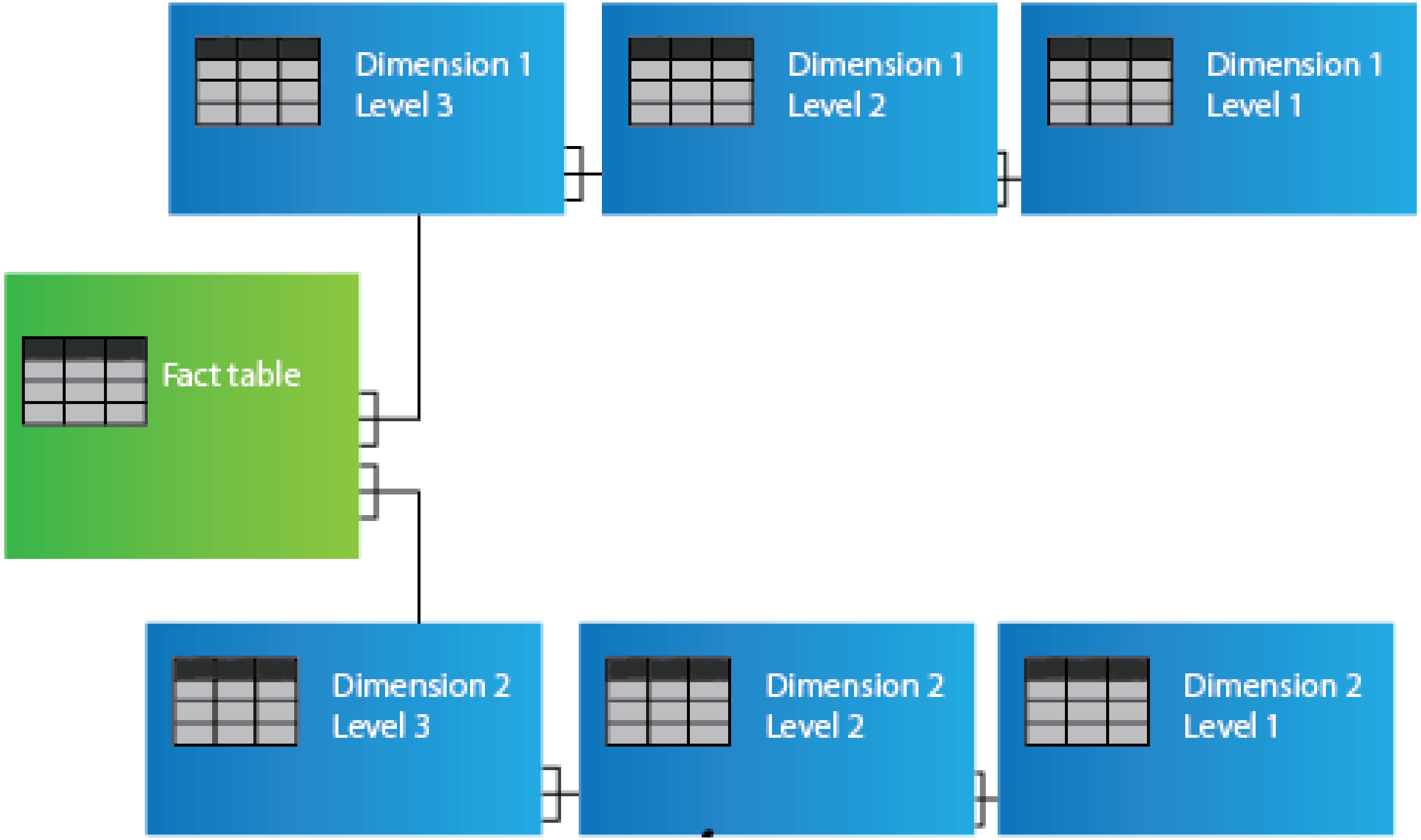
Characteristics of Snowflake Schema

- The main benefit of the snowflake schema it uses smaller disk space.
- Easier to implement a dimension is added to the Schema
- Due to multiple tables query performance is reduced
- The primary challenge that you will face while using the snowflake Schema is that you need to perform more maintenance efforts because of the more lookup tables.

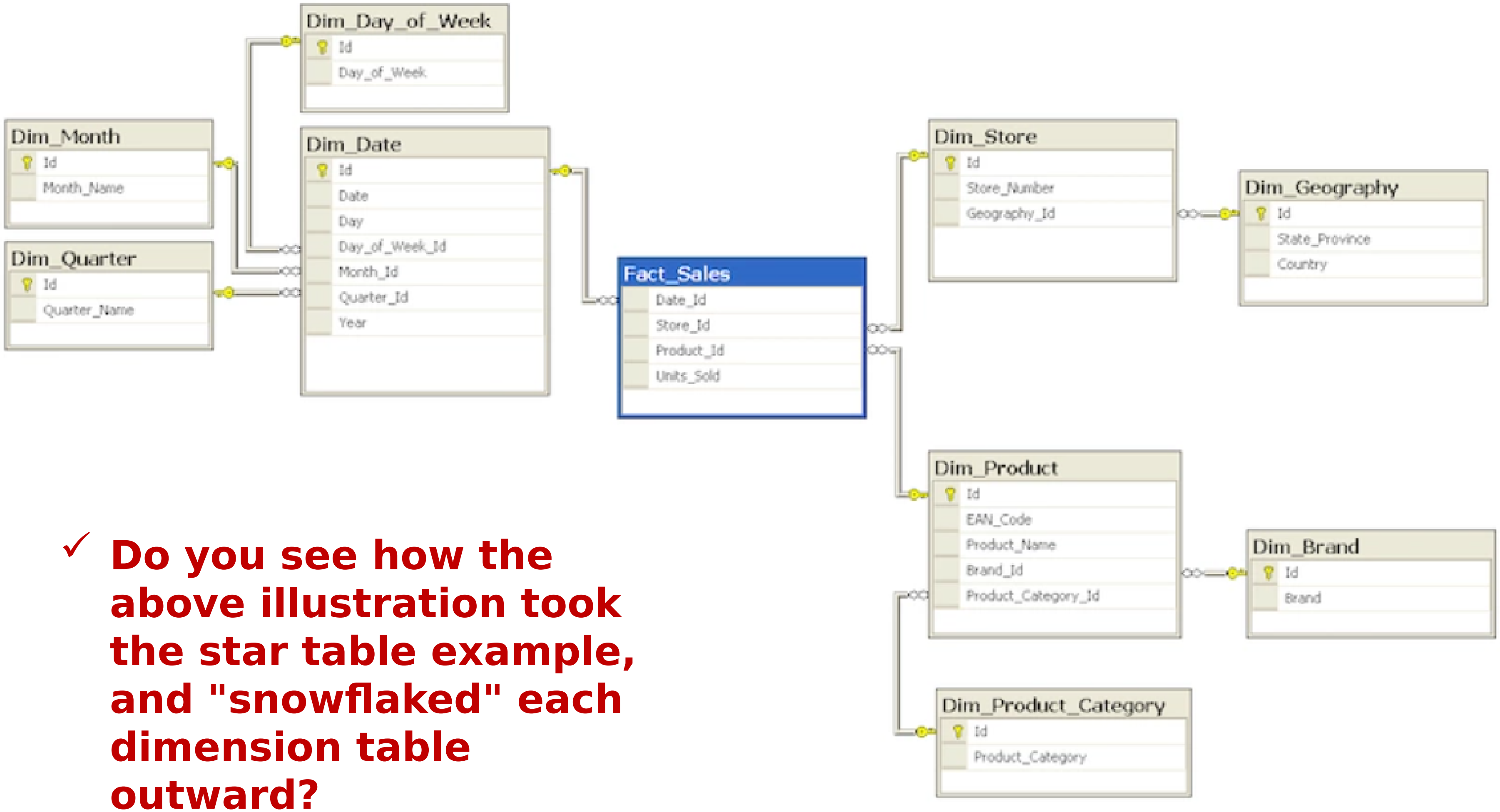
- Vertabelo offers an excellent comparison of snowflake schemas versus star schemas:

  "***Unlike the star schema, dimension tables in the snowflake schema can have their own categories. The ruling idea behind the snowflake schema is that dimension tables are completely normalized. Each dimension table can be described by one or more lookup tables. Each lookup table can be described by one or more additional lookup tables. This is repeated until the model is fully normalized. The process of normalizing star schema dimension tables is called snowflaking***."

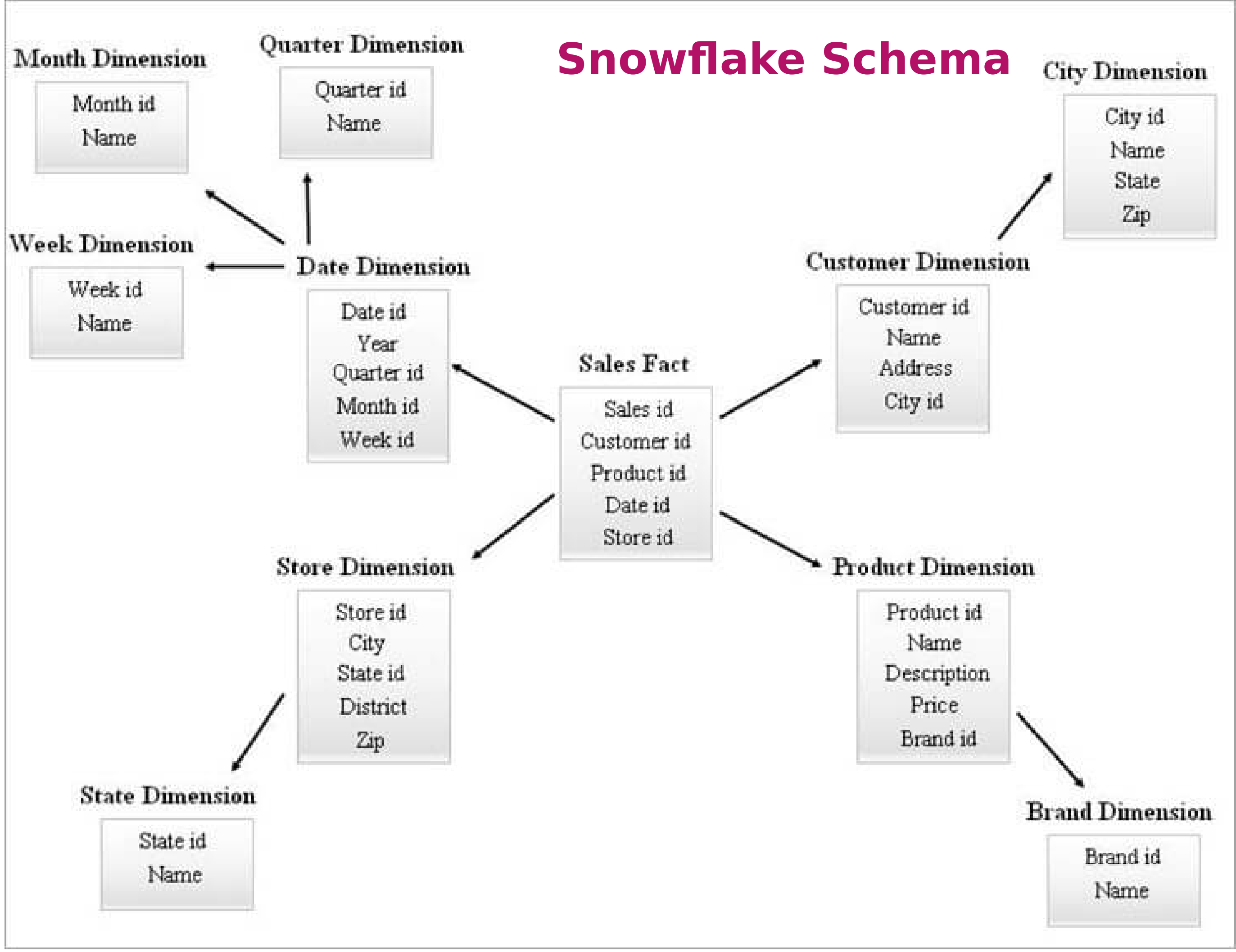




Snowflake Schema



- ✓ **Do you see how the above illustration took the star table example, and "snowflaked" each dimension table outward?**
- ✓ Let's examine the Dim_Product dimension table.



Snowflake Schema

Month Dimension
Month id
Name

Quarter Dimension
Quarter id
Name

City Dimension
City id
Name
State
Zip

Week Dimension
Week id
Name

Date Dimension
Date id
Year
Quarter id
Month id
Week id

Customer Dimension
Customer id
Name
Address
City id

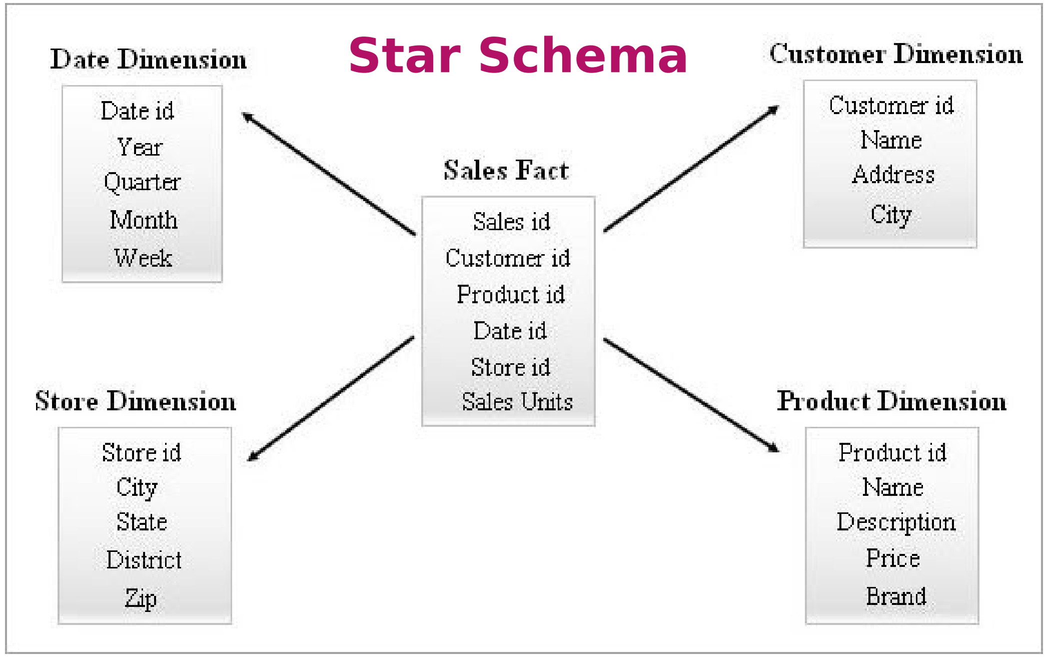
Sales Fact
Sales id
Customer id
Product id
Date id
Store id

Store Dimension
Store id
City
State id
District
Zip

Product Dimension
Product id
Name
Description
Price
Brand id

State Dimension
State id
Name

Brand Dimension
Brand id
Name



Star Schema

Date Dimension
Date id
Year
Quarter
Month
Week

Customer Dimension
Customer id
Name
Address
City

Sales Fact
Sales id
Customer id
Product id
Date id
Store id
Sales Units

Store Dimension
Store id
City
State
District
Zip

Product Dimension
Product id
Name
Description
Price
Brand

Benefits of Snowflake Schemas

- **Compatible with many OLAP database modelling tools**: Certain OLAP database tools, which data scientists use for data analysis and modelling, are specifically designed to work snowflake data schemas.
- **Saves on data storage requirements**: Normalizing the data that would typically get denormalized in a star schema can offer a tremendous reduction in disk space requirements.
  - Essentially, this is because you're converting long strings of non-numerical data (the information pertaining to descriptors and names) into numerical keys that are dramatically less taxing from a storage perspective.

Challenges of Snowflake Schemas

- **Complex data schemas**: As you might imagine, snowflake schemas create many levels of complexity while normalizing the attributes of a star schema.
  - This complexity results in more complicated source query joins.
  - In offering a more efficient way to store data, snowflake can result in performance declines while browsing these complex joins.
  - Still, processing technology advancements have resulted in improved snowflake schema query performance in recent years, which is one of the reasons why snowflake schemas are rising in popularity.
- **Slower at processing cube data**: In a snowflake schema, the complex joins result in slower cube data processing.
  - The star schema is generally better for cube data processing.
- **Lower data integrity levels**: While snowflake schemas offer greater normalization and fewer risks of data corruption after performing UPDATE and INSERT commands, they do not provide the level of transnational assurance that comes with a traditional, highly-normalized database structure.
  - Therefore, when loading data into a snowflake schema, it's vital to be careful and double-check the quality of information post-loading.

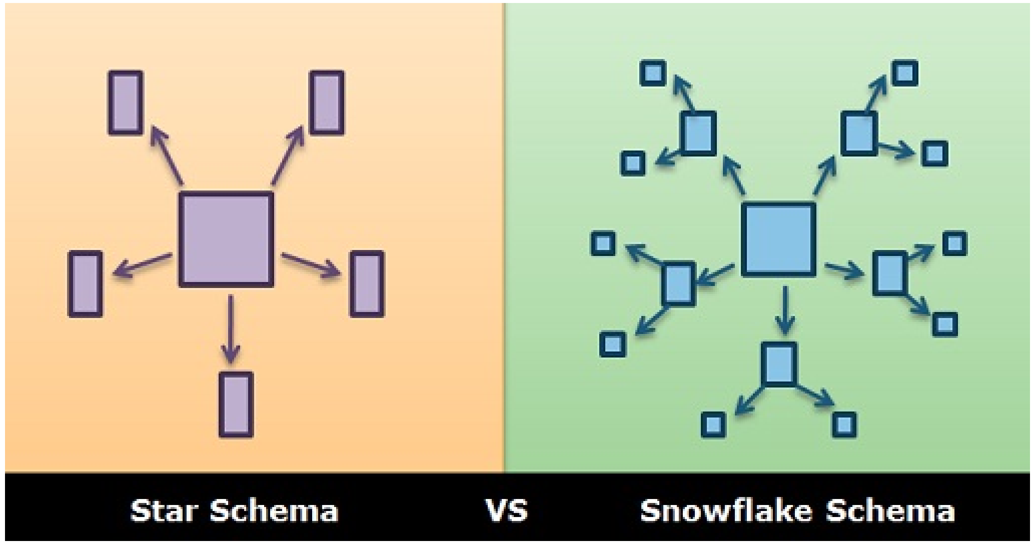
The steps involved in designing the Star Schema

1. Identification of the business process for analysis (for example sales)
2. Identification of the measures or facts (sales currency)
3. Identification of dimensions for facts (products, locations, time and company dimensions)
4. Listing of the columns which describe each dimension (city name, region name, branch name, branch code)
5. Determining the lowest level of summary in a fact table (sales currency)

- **Important aspects of Star Schema & Snow Flake Schema**:
  - In a star schema every dimension will have a primary key.
  - In a star schema, a dimension table will not have any parent table.
  - Whereas in a snow flake schema, a dimension table will have one or more parent tables.
  - Hierarchies for the dimensions are stored in the dimensional table itself in star schema.
  - Whereas hierarchies are broken into separate tables in snow flake schema. These hierachies helps to drill down the data from topmost hierarchies to the lowermost hierarchies.

Star Schema vs. Snowflake Schema: 5 Critical Differences

- Star schema dimension tables are not normalized, snowflake schemas dimension tables are normalized.
- Snowflake schemas will use less space to store dimension tables but are more complex.
- Star schemas will only join the fact table with the dimension tables, leading to simpler, faster SQL queries.
- Snowflake schemas have no redundant data, so they're easier to maintain.
- Snowflake schemas are good for data warehouses, star schemas are better for datamarts with simple relationships.



Star Schema Vs Snowflake Schema: Key Differences

Star Schema	Snowflake Schema
Hierarchies for the dimensions are stored in the dimensional table.	Hierarchies are divided into separate tables.
It contains a fact table surrounded by dimension tables.	One fact table surrounded by dimension table which are in turn surrounded by dimension table
In a star schema, only single join creates the relationship between the fact table and any dimension tables.	A snowflake schema requires many joins to fetch the data.
Simple DB Design.	Very Complex DB Design.
Denormalized Data structure and query also run faster.	Normalized Data Structure.
High level of Data redundancy	Very low-level data redundancy
Single Dimension table contains aggregated data.	Data Split into different Dimension Tables.
Cube processing is faster.	Cube processing might be slow because of the complex join.
Offers higher performing queries using Star Join Query Optimization. Tables may be connected with multiple dimensions.	The Snowflake schema is represented by centralized fact table which unlikely connected with multiple dimensions.

Summary

- Star and Snowflake schema is used for designing the data warehouse.
- Both have certain merits and demerits where snowflake schema is easy to maintain, lessen the redundancy hence consumes less space but complex to design.
- Whereas star schema is simple to understand and design, uses less number of joins and simple queries but have some issues such as data redundancy and integrity.
- The star schema is the simplest type of Data Warehouse schema. It is known as star schema as its structure resembles a star.
- A Snowflake Schema is an extension of a Star Schema, and it adds additional dimensions. It is called snowflake because its diagram resembles a Snowflake. The dimension tables are normalized which splits data into additional tables.
- In a star schema, only single join defines the relationship between the fact table and any dimension tables where as a snowflake schema requires many joins to fetch the data.
- Star schema contains a fact table surrounded by dimension tables. It is also known as Star Join Schema and is optimized for querying large data sets.
- Snowflake schema is surrounded by dimension table which are in turn surrounded by dimension table