Databases & Data Stores

Terminology used in describing databases

ACID

- Atomicity (a database **transaction either succeeds or fails**)
- Consistency (use of rules and constraints so **state is always valid**)
- Isolation (**transactions happen in isolation**)
- Durability (once committed the **transaction is permanen**t/ stored in non-volatile storage like a hard disk)

BASE

- Basically Available (generally available for queries; **no isolation/waiting for other transactions**)
- Soft State (because consistency is *eventual*, **state may show some lag**)
- Eventually Consistent (as a system's **state is gradually replicated across all nodes**, it eventually becomes consistent.

CAP Theorem **(architectural goal is all three, distributed networks reality - choose two)**

- Consistency – no updates to any node if nodes can't communicate (nodes stay in sync)
- Availability – system always responds
- Partition Tolerance – replication across nodes

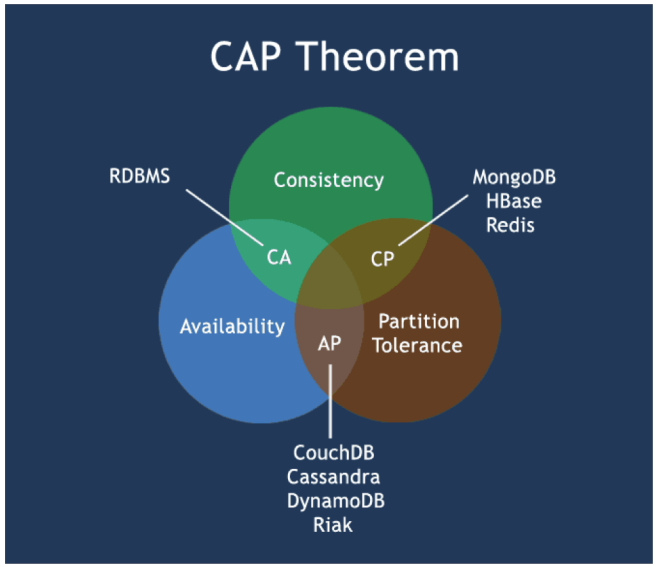


Image: https://practice.geeksforgeeks.org/problems/cap-theorem

RDBMS / SQL Database / Transactional Database

- Most commonly used structure in enterprise scenarios
- Built-in data integrity
- Support ACID compliance to guarantee data validity
- Difficult to scale (unless sharding techniques are used)
  - Sharding is the separation of very large databases into smaller, faster, more easily managed parts

Entity Relationship Diagram (ERD)

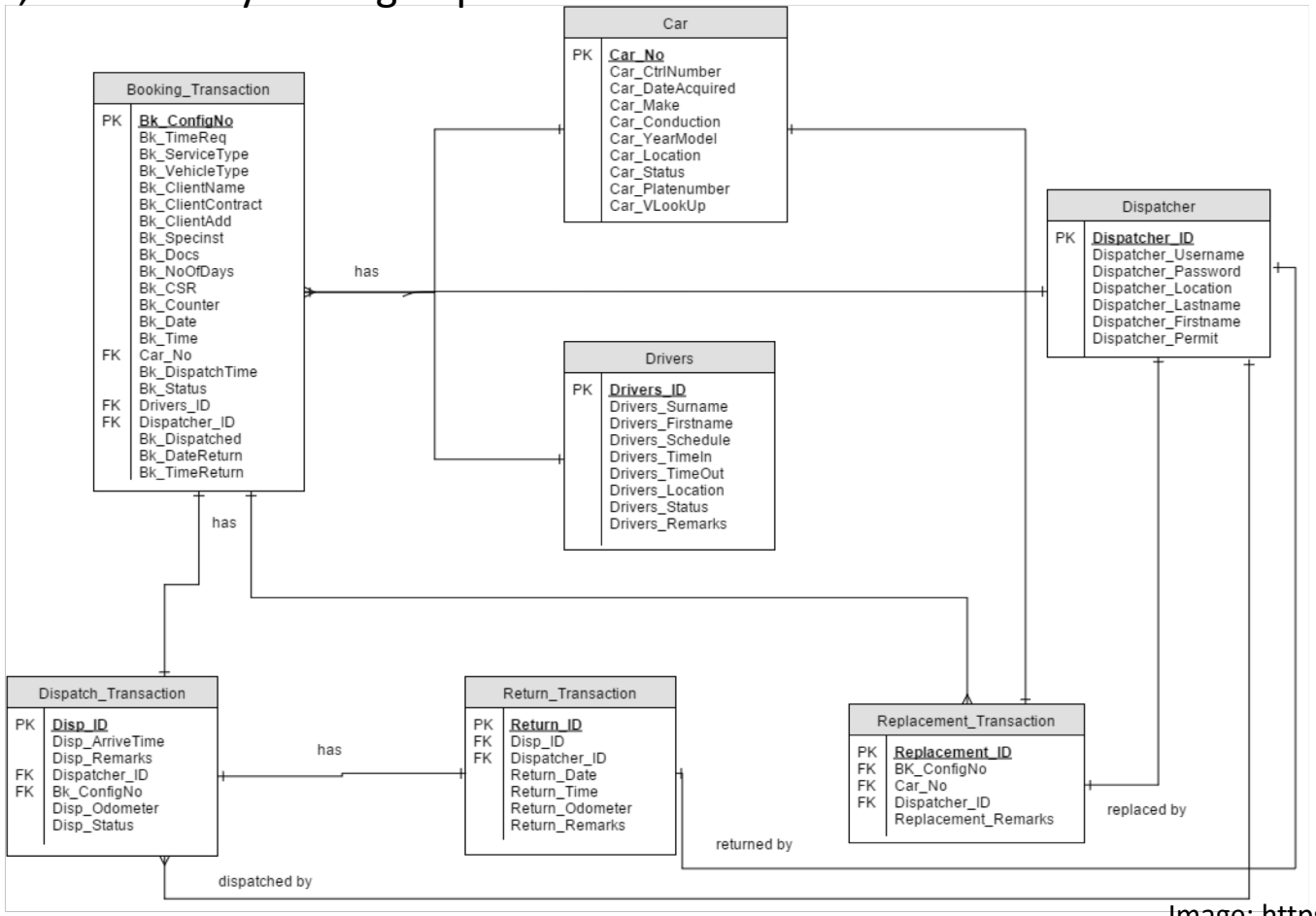


Image: https://sites.google.com/site/dracvbms/entity-relationship-diagram

NoSQL – Not only SQL (structured query language)

NoSQL term coined by Carlo Strozzi in 1998 to describe his relational database that did not expose Structured Query Language to users.

What we mean by NoSQL is different (non-relational database), and that term came into use for that meaning around 2009.

Four main types

1. **Key-Value Store**
   - Riak, Amazon S3 (Dynamo), Redis, Voldemort, Apache Ignite, Dynamo, etc.
2. **Document Store**
   - Apache CouchDB, Couchbase, RethinkDB, IBM Domino, CosmosDB, MongoDB, etc.
3. **Column Store**
   - Hbase, Cassandra, BigTable, Vertica, Accumulo, Druid, etc.
4. **Graph**
   - Neo4J, Apache Giraph, AllegroGraph, Virtuoso, InfiniteGraph, FlockDB, etc.

Data Model	Performance	Scalability	Flexibility	Complexity	Functionality
Key–Value Store	high	high	high	none	variable (none)
Column-Oriented Store	high	high	moderate	low	minimal
Document-Oriented Store	high	variable (high)	high	low	variable (low)
Graph Database	variable	variable	high	high	graph theory
Relational Database	variable	variable	low	moderate	relational algebra

Key-Value Store

- Riak, Amazon S3 (Dynamo), Redis, Voldemort, Apache Ignite, Dynamo, etc.
- Data is represented as a collection of key-value pairs, with each key appearing only once in a collection
- Computationally powerful Key-Value stores may keep keys in lexicographic order, enabling faster retrieval.

Key	Value
Title:How to X	by Jane Doe. Categories: foo, bar. So you want to X? Here's ...
Title:X Y About Z	by John Doe. Here's a list of Y about Z...
Title:Why X Instead of Z	by Jane Doe. Categories: foo. Some people think Z, but...

Image: https://readwrite.com/2011/08/12/from-big-data-to-nosql-the-rea-2/

Document Store

- Apache CouchDB , MongoDB, Couchbase, RethinkDB, IBM Domino, CosmosDB, etc.
- Data is represented as a map
- Like a key-value store, but the value is a document (XML, JSON, BSON, etc.)
- Documents are self-describing and hierarchical trees

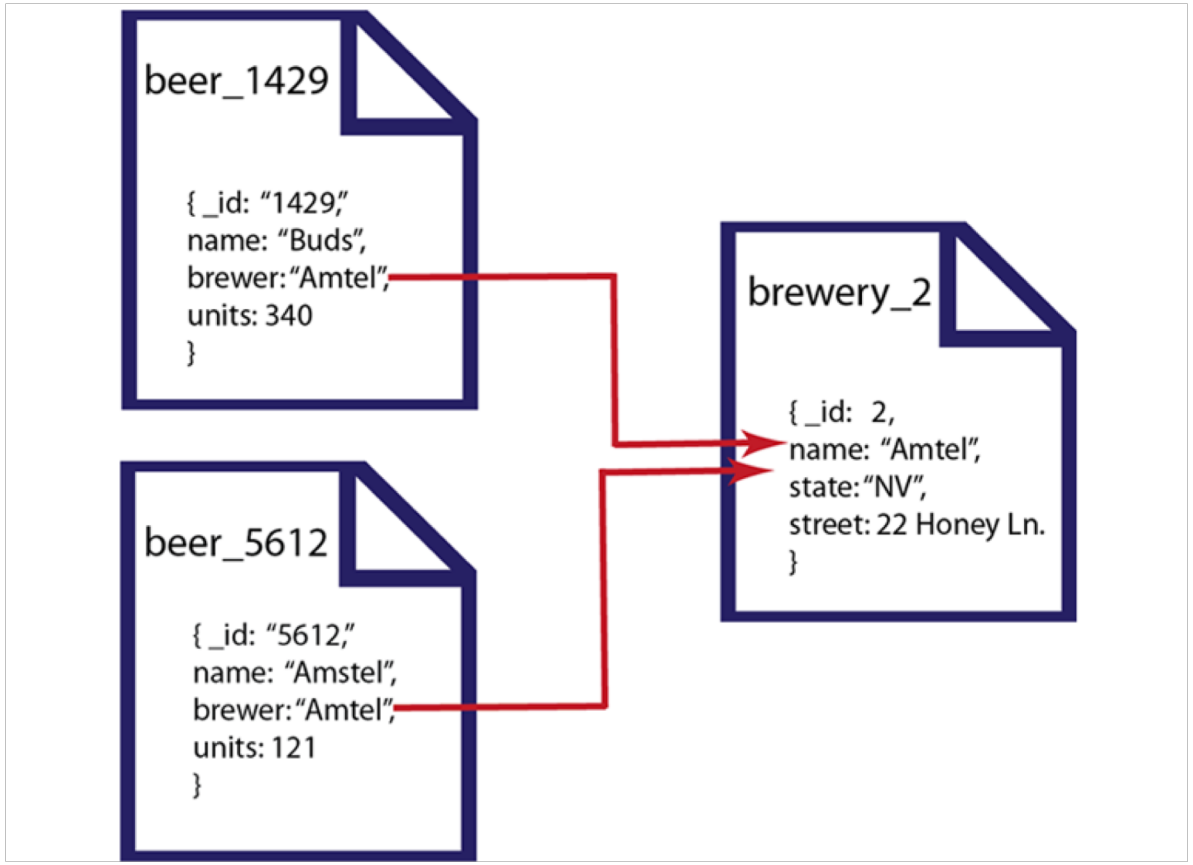
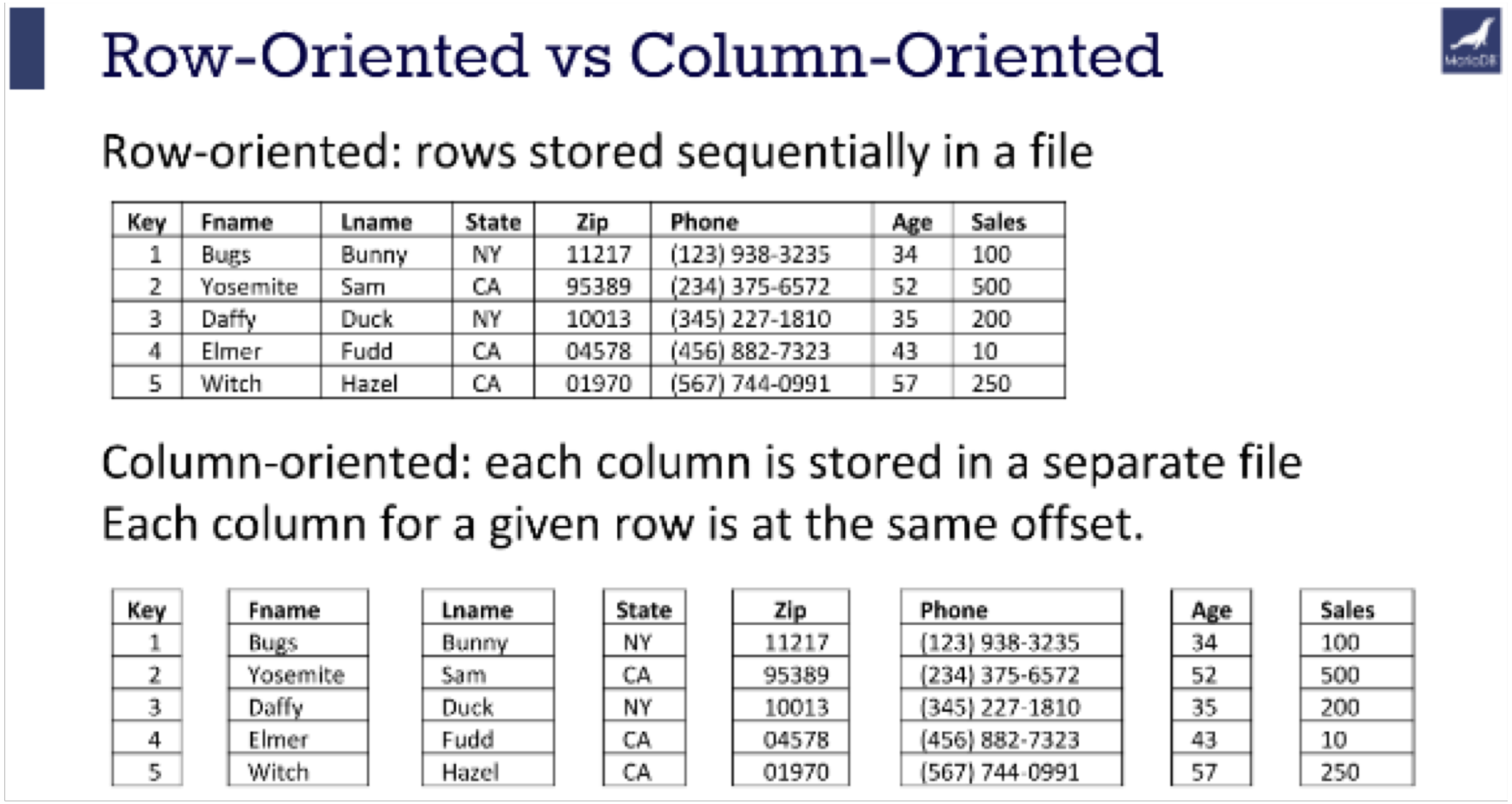


Image: http://amanbinny.blogspot.com/2014/11/introduction-about-nosql-data-models.html

Column Store

- Hbase, Cassandra, MariaDB, BigTable, Vertica, Accumulo, Druid, etc.
- All cells corresponding to a column are stored as a continuous disk entry, making the search/access faster.
- Optimized for reads (not so much for create, update, or delete)



Key	Fname	Lname	State	Zip	Phone	Age	Sales
1	Bugs	Bunny	NY	11217	(123) 938-3235	34	100
2	Yosemite	Sam	CA	95389	(234) 375-6572	52	500
3	Daffy	Duck	NY	10013	(345) 227-1810	35	200
4	Elmer	Fudd	CA	04578	(456) 882-7323	43	10
5	Witch	Hazel	CA	01970	(567) 744-0991	57	250

Key	Fname	Lname	State	Zip	Phone	Age	Sales
1	Bugs	Bunny	NY	11217	(123) 938-3235	34	100
2	Yosemite	Sam	CA	95389	(234) 375-6572	52	500
3	Daffy	Duck	NY	10013	(345) 227-1810	35	200
4	Elmer	Fudd	CA	04578	(456) 882-7323	43	10
5	Witch	Hazel	CA	01970	(567) 744-0991	57	250

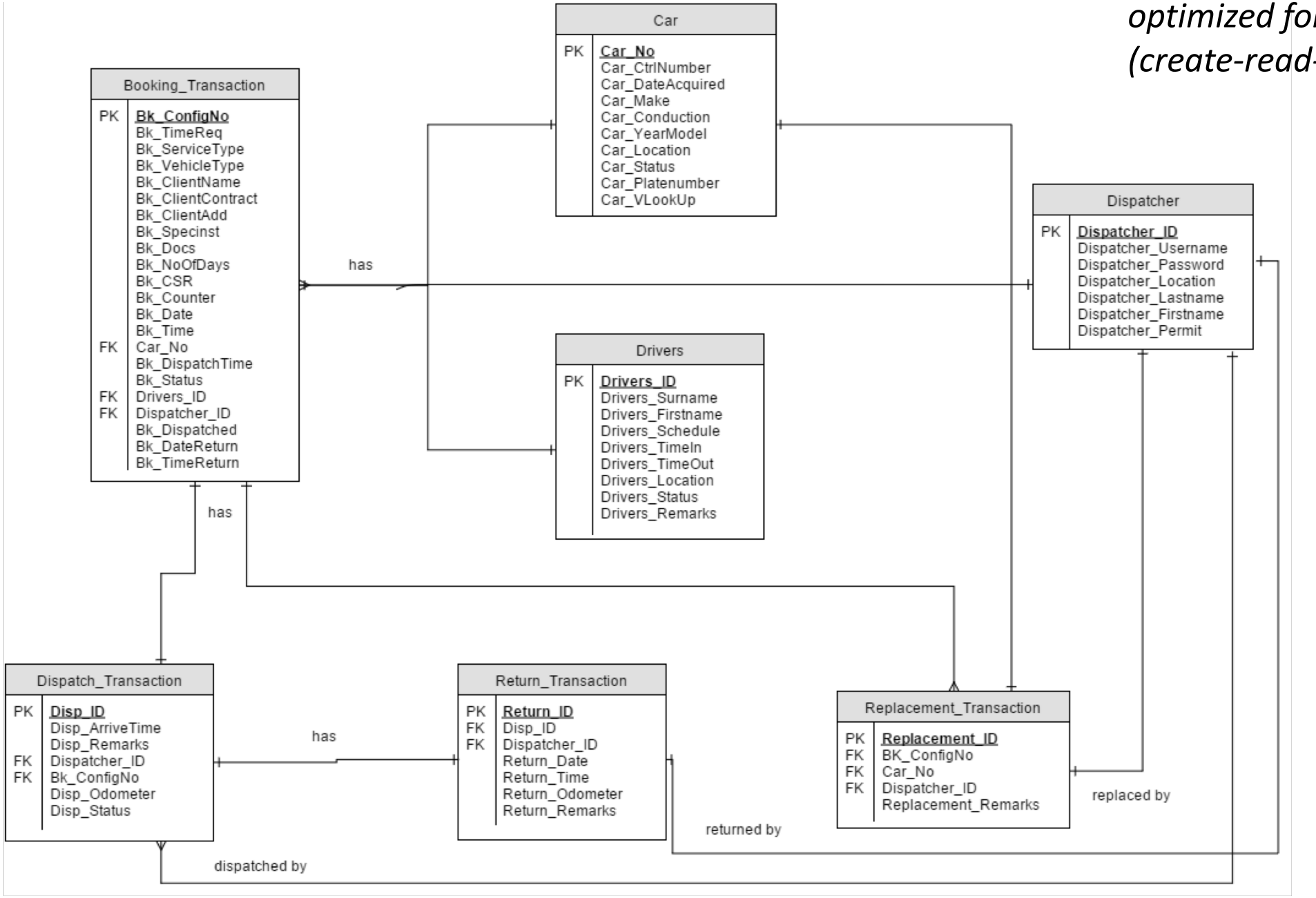
Image: https://mariadb.com/resources/blog/why-columnstore-important

OLTP: OnLine Transactional Processing

- Should respond immediately to requests
- High throughput and insert/update intensive
- Used by many users concurrently
- Key goals: availability, speed, concurrency, and recoverability

normalized relational database

optimized for CRUD transactions (create-read-update-delete)

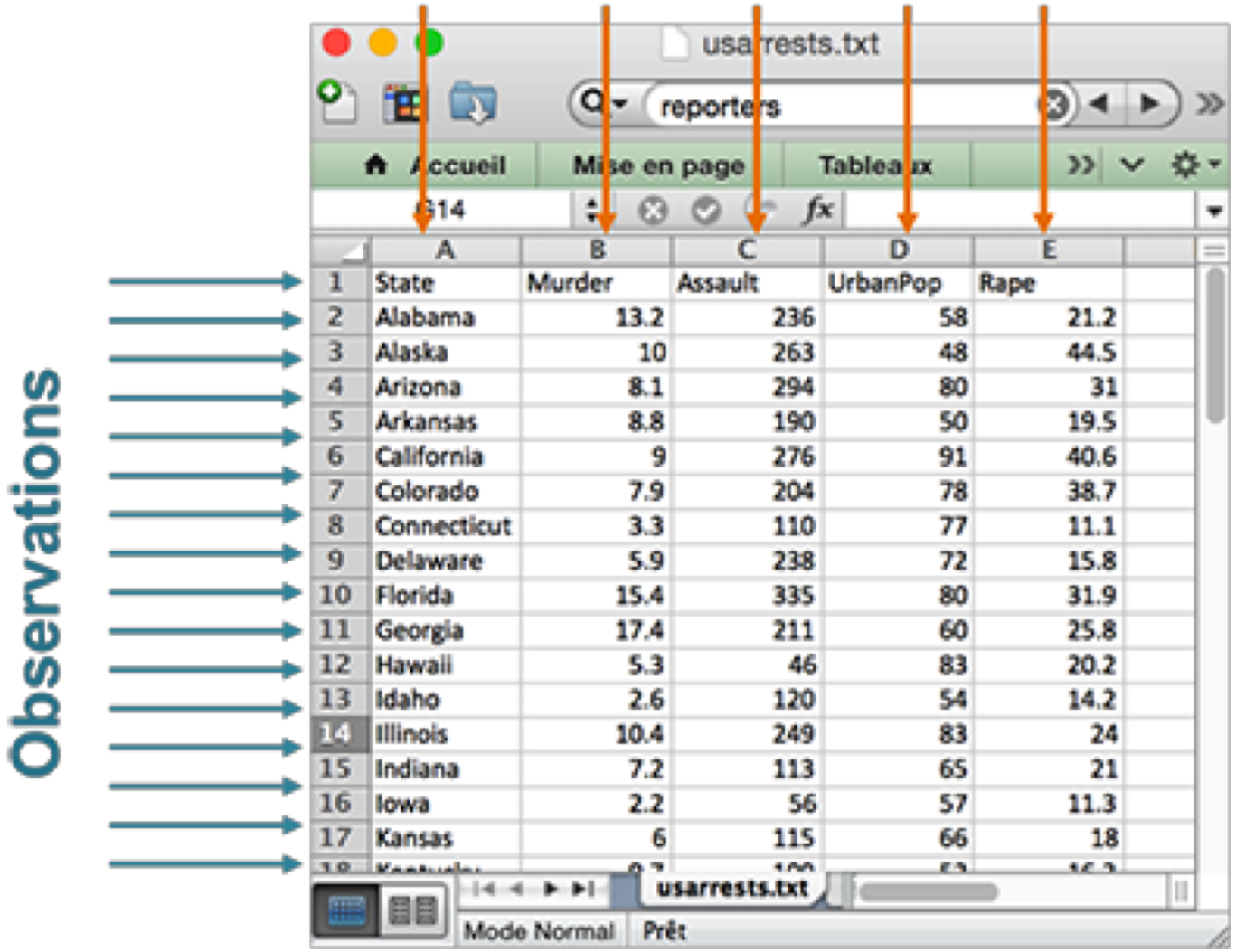


Referential integrity

- Constrains the possible values for a given field
- Lookup tables (sometimes called code tables) have a primary key that is used as a foreign key in the transaction table

Tidy data

tidy data – *optimized for ad hoc analyses*
(data merged and flattened)

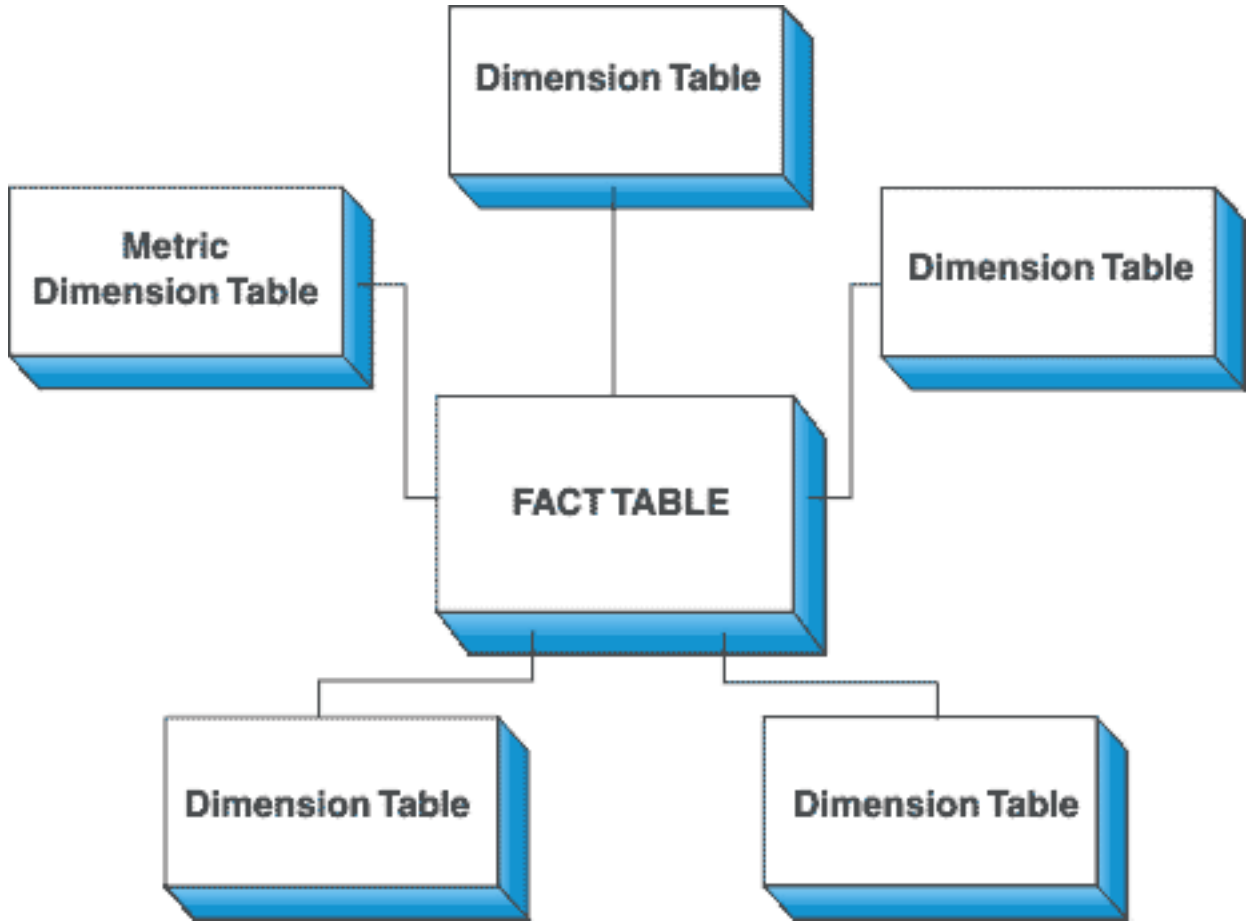


	A	B	C	D	E
1	State	Murder	Assault	UrbanPop	Rape
2	Alabama	13.2	236	58	21.2
3	Alaska	10	263	48	44.5
4	Arizona	8.1	294	80	31
5	Arkansas	8.8	190	50	19.5
6	California	9	276	91	40.6
7	Colorado	7.9	204	78	38.7
8	Connecticut	3.3	110	77	11.1
9	Delaware	5.9	238	72	15.8
10	Florida	15.4	335	80	31.9
11	Georgia	17.4	211	60	25.8
12	Hawaii	5.3	46	83	20.2
13	Idaho	2.6	120	54	14.2
14	Illinois	10.4	249	83	24
15	Indiana	7.2	113	65	21
16	Iowa	2.2	56	57	11.3
17	Kansas	6	115	66	18

OLAP: OnLine Analytical Processing

- Longer running tasks
- More complicated queries against larger volumes of data
- Precomputed values can be stored as OLAP cubes (3-dimensions)
  - For dimensions like:
    - Dates (year, quarter, month)
    - Geographic (country, city, region)
    - Product Category
    - Customer
  - Aggregated totals are pre-calculated and stored at the “intersections” of the cube
- OLAP Operations
  - ***Slicing*** (holding the value of one dimension fixed, like looking at one particular year of sales data)
  - ***Dicing*** (looking at a subset of each dimension, like looking at sales between 2015 and 2017 for the Nashville and Memphis markets)
  - ***Rollup*** (summarizing/aggregating the data along a dimension, like calculating the total sales for 2016)
- Data sometimes stored as columnar data stores

star schema model (AKA dimensional database)



A database model optimized for queries and data warehouse tools.

Goal: query performance and schema simplicity

Design: “fact table” surrounded by some number of “dimension tables”

Star Schema

Facts:

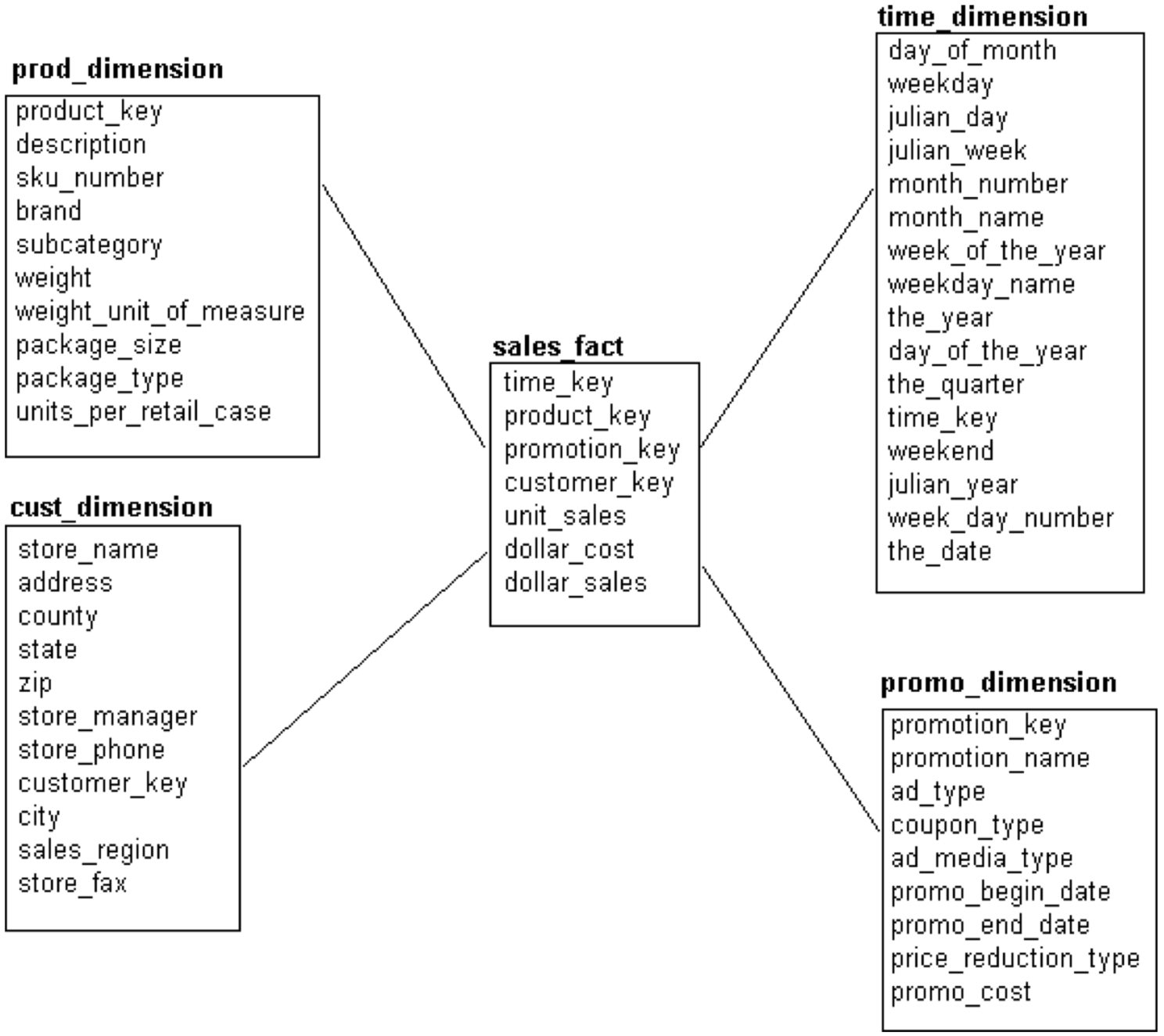
- Usually numeric measurements of a business
- Examples: Sales dollars, sales units, costs
- Often one fact per business transaction
- Change regularly

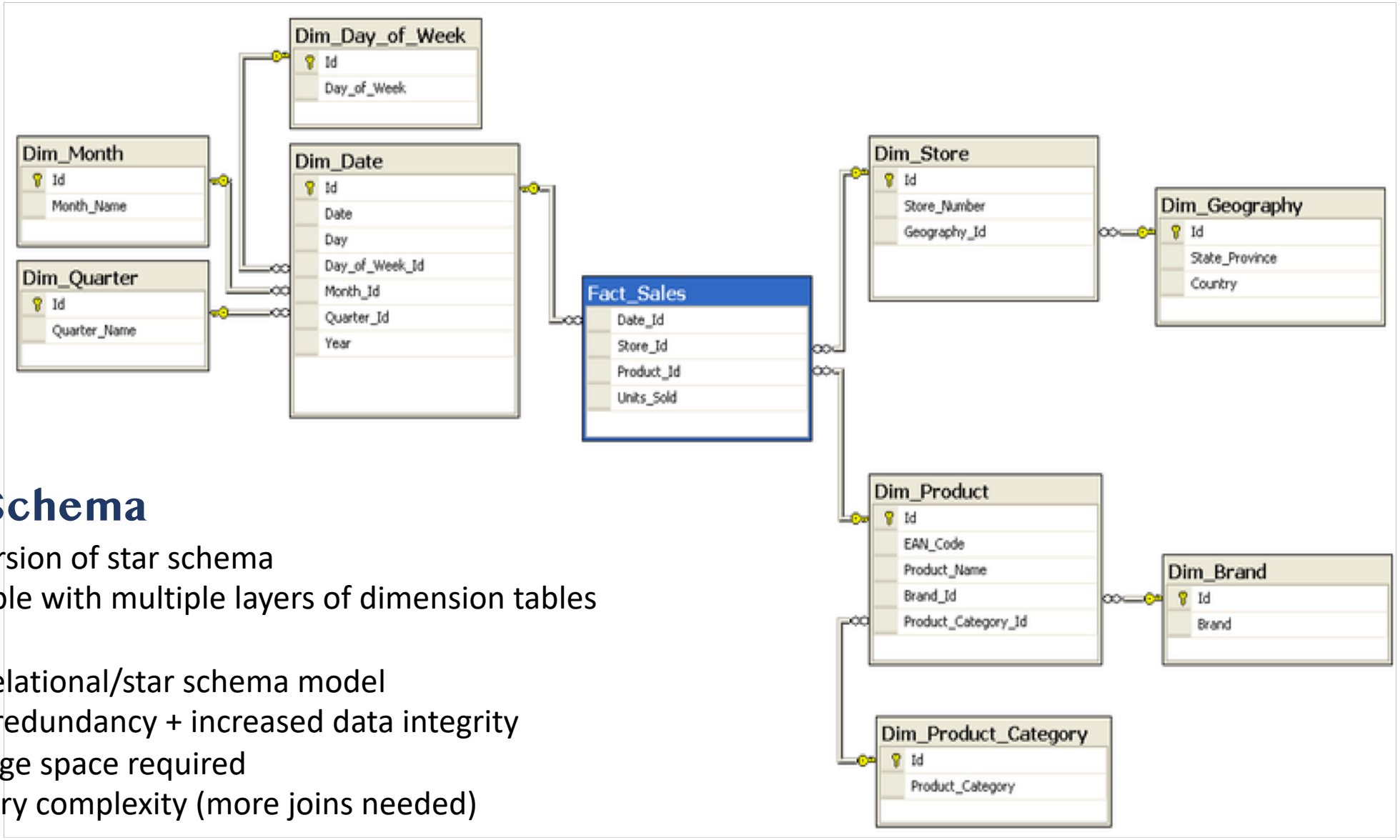
Dimensions:

- Descriptive data providing context for facts
- Examples: Product description, customer contact information, time dimensions
- Rarely if ever change

Star Schema

- Fact table is connected to dimensional tables through keys
- The primary key of each dimension table is linked to a foreign key of fact table
- Allows fast queries



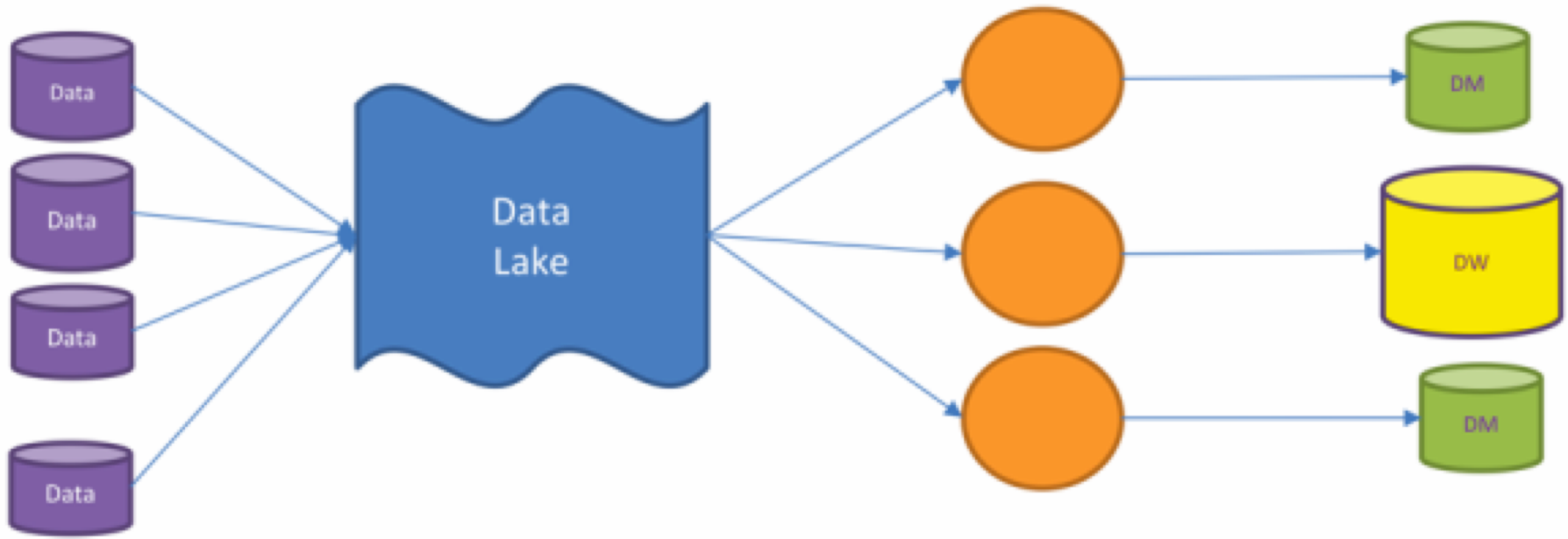


Snowflake Schema

- Normalized version of star schema
- Central fact table with multiple layers of dimension tables

- Like a hybrid relational/star schema model
  - Reduced redundancy + increased data integrity
  - Less storage space required
  - More query complexity (more joins needed)

The Data Lake Pattern



Data Sources

Data in raw format

Data transforms

Data ready for each need