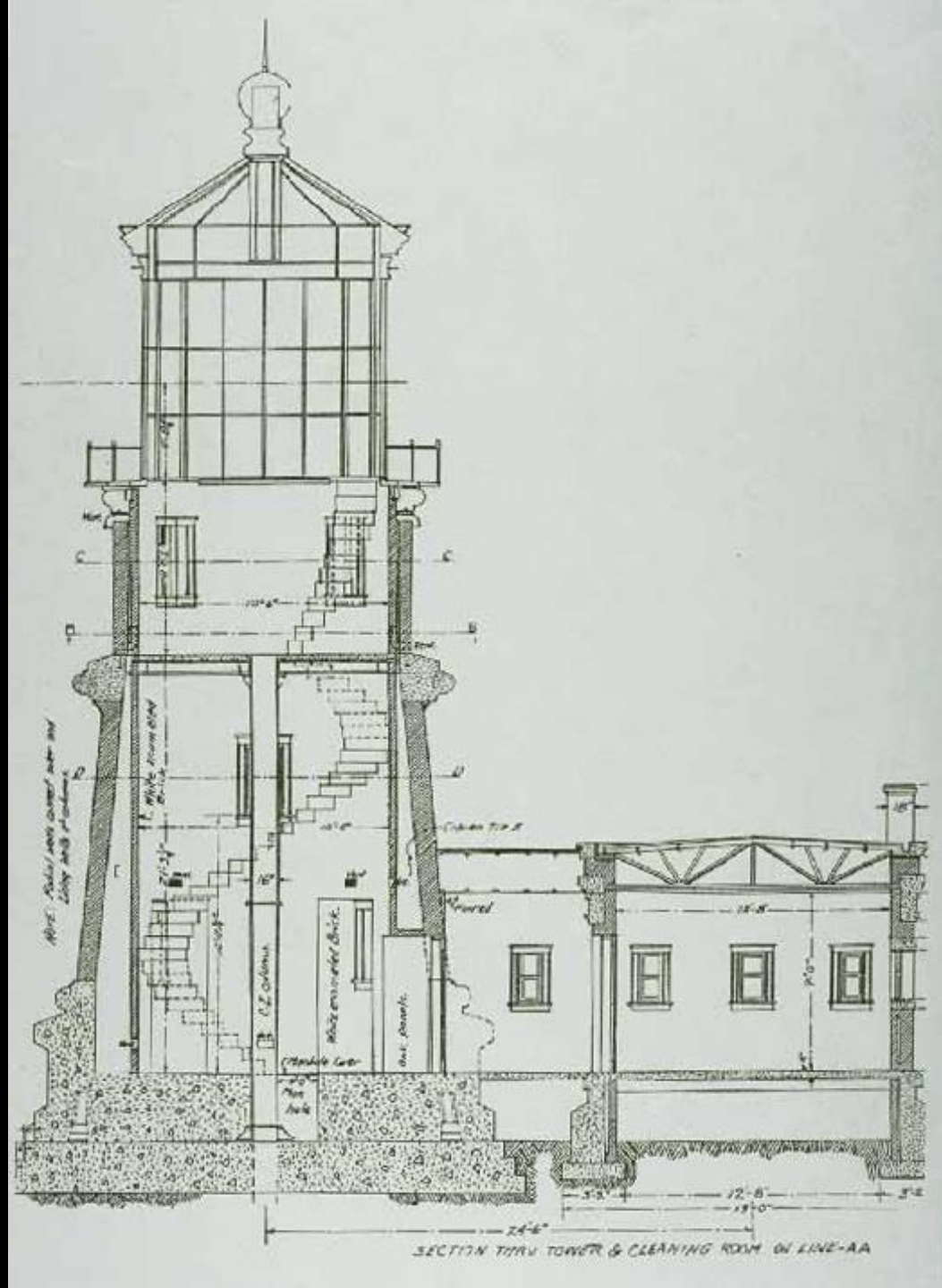
Software Architecture & Design



Databases & Persistency

The Persistency Layer of Client/Server

Contents

- History & Context: How the RDBMS emerged to dominate
- Database Architecture
- Distributed Databases

History & Context

How database technology evolved & why server DBMS dominate

Early Years – Hierarchical Database

- Mainframe Computing
  - File based systems
  - Batch processing
    - 'Jobs' consisting of hundreds/thousands of input/output files
    - Typically run overnight
    - Overnight updates/reports for next day of business
- Hierarchical & Networked databases
  - Developed by IBM '60s
  - Used in Mainframes
  - E.g. Information Management System (IMS)
  - https://en.wikipedia.org/wiki/Hierarchical_database_model

https://en.wikipedia.org/wiki/IBM#/media/File:IBM_logo.svg

Relational Database

- IBM – System-R (mid ‘70s) →IBM DB2 ’83 Mainframe

- Oracle – 1979
- Introduces abstraction over physical implementation
- Performance & Scalability
  - Slower than hierarchical
  - Capacity less than hierarchical

Relation Databases – the ‘Big 4’

(circa early ‘90s)

Relation Databases – new players

(circa mid ‘90s)

Relation Databases – new players

(circa mid 2000s)

Acquisitions & Competition

- Informix acquired by IBM in 2001
- Sybase was acquired by SAP in 2010
  - Ceased using the Sybase name in 2014.
- Ingres competed head-to-head with Oracle in early ‘80s
  - Started to lose market share from ‘85 onwards
- Oracle acquired MySQL.com
  - MariaDB soon emerged

Relation Databases – Changing Landscape

RDBMS & Platforms – 90's

Mainframe

Server

Desktop & Network

RDBMS & Platforms – 90's

IBM DB/2

Tandem NS-SQL

Mainframe

Sybase

Ingres

Server

Oracle

Informix

MS SQL-Server

dBase III/dBase IV

FoxBase (xBase)

Paradox

Desktop & Network

Access

DataEase

RDBMS & Platforms – Now

Mainframe

IBM DB/2

MySQL/MariaDB

Server

Oracle

MS SQL-Server

Desktop & Network

Access

Windows - Unix/Linux

Alternative Databases – OO & DMS

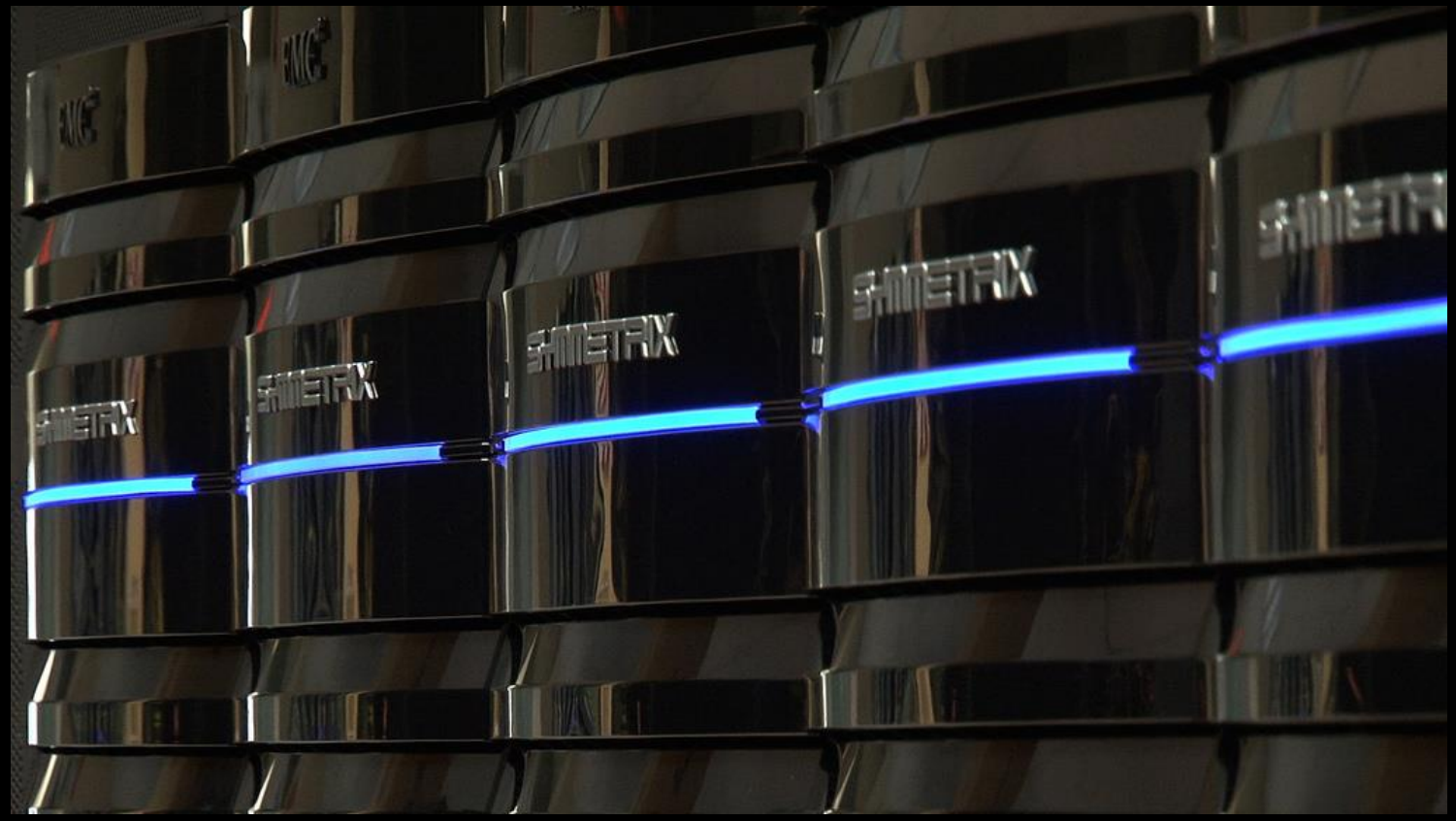
- Prevalence of OO development called for Object-Oriented Databases
  - E.g. O2
  - Largely unable to establish themselves (to a position of dominance)
- Document Management Systems emerge – circa ‘90s
  - Used to store documents, including XML, SGML, HTML
  - Good candidates for object persistence
- ORM – Object Relational Mappers
  - Abstraction Layer within server
  - E.g.
    - Postgres – an open-source object-relational DBMS
    - Hibernate

Influences: BPR

- Business Process Re-engineering
  - TQM for the office (Total Quality Management)
- Client/Server revolution allowed computing power & resources to be placed closer to where decisions needed to be made
- Enabled corporations to rethink their organizational structure
- Where is the data?
  - Move the data closer to the decision making.

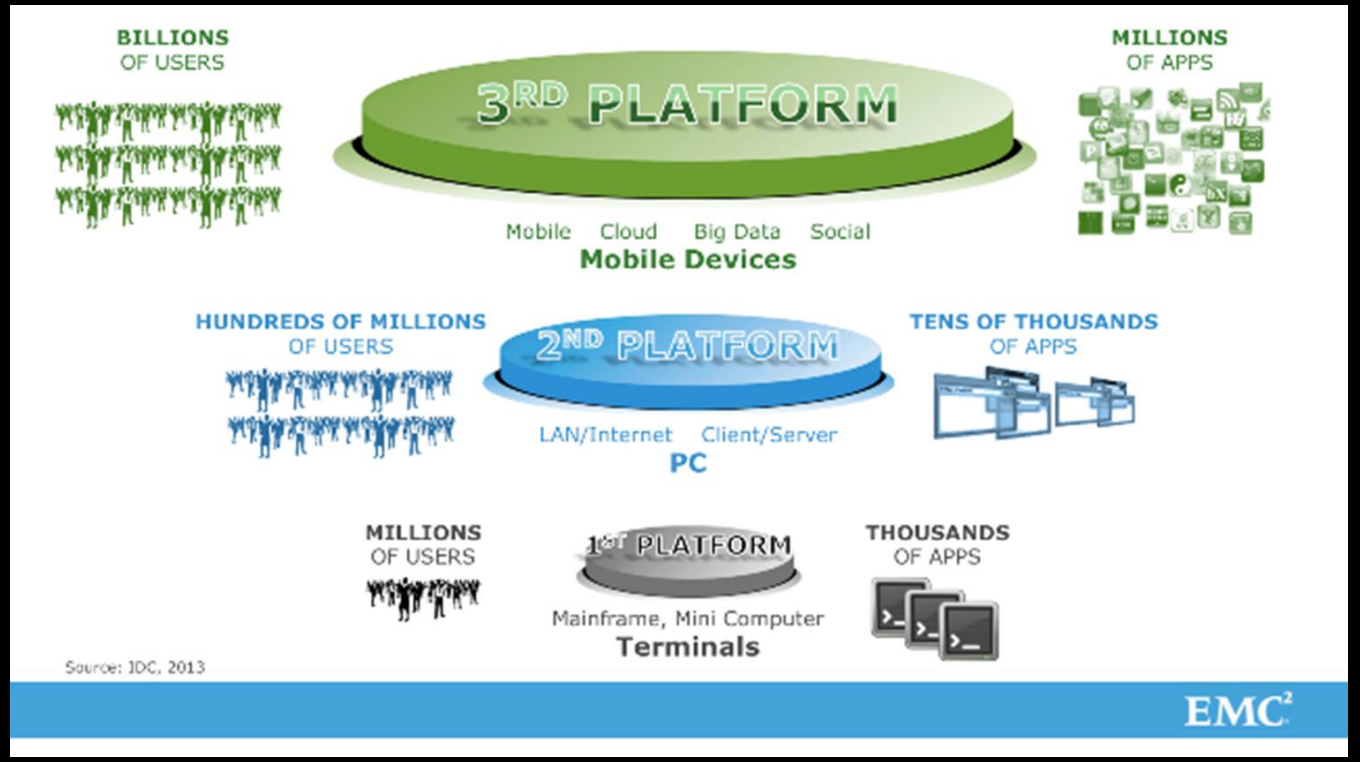
Influences - Databases and SAN

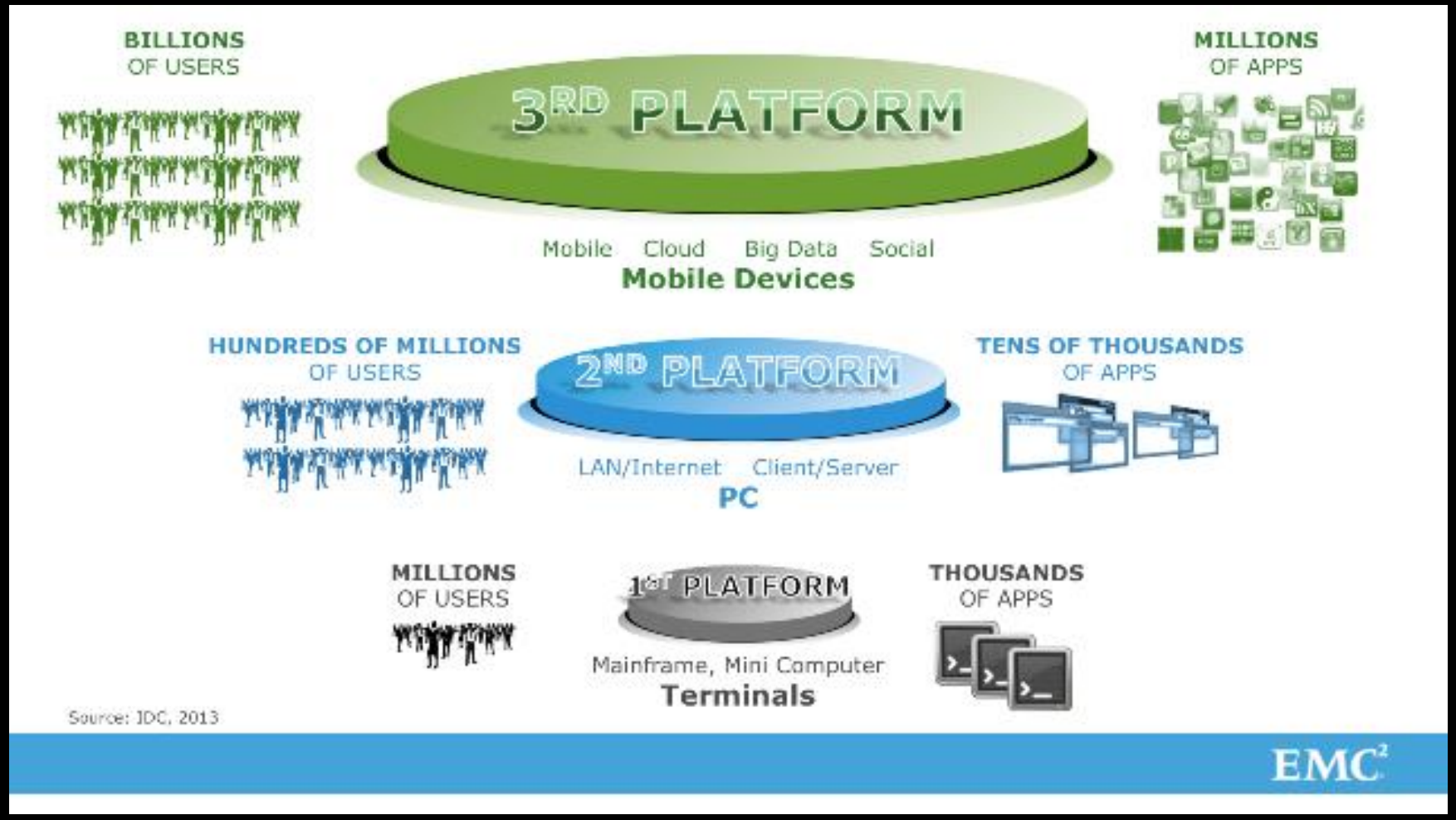
- High Performant, Sophisticated NAS
- Enterprise-level Storage Arrays
  - Mirror & RAID
  - Remote Replication (RDF)
  - Full Management



Influences - The Third Platform

- Distinctions
  - Number of Users
  - Number of Devices
  - Number of Applications
- 4 Pillars
  - Cloud
  - Big Data
  - Social
  - Mobile
- Microservices Architecture
  - Characterized by Multiple Persistency Mechanisms





THE THIRD PLATFORM

The Third Platform is described by IDC as the next-generation compute platform that is accessed from mobile devices, utilizes Big Data, and is cloud based.

BILLIONS OF USERS

3RD PLATFORM

Mobile Big Data Social

CLOUD

MILLIONS OF APPS

HUNDREDS OF MILLIONS OF USERS

2ND PLATFORM

LAN/Internet Client/Server

DISTRIBUTED

TENS OF THOUSANDS OF APPS

MILLIONS OF USERS

1ST PLATFORM

Mainframe, Mini Computer

MAINFRAMES

THOUSANDS OF APPS

Alternative Databases – Internet Era

- Proliferation of non-structured data – 2000 onwards
  - E.g. Internet Companies - Social
    - e-mail, Photos, Videos, Posts, Messages & IM Conversations
    - Non-transactional, always appending, never editing
    - Not necessarily suited to relational model
  - Need for alternative persistency engines
    - E.g.
    - MongoDB – an open source document-oriented database
    - Redis – an open-source, networked, in-memory key-value data store

Databases at Scale

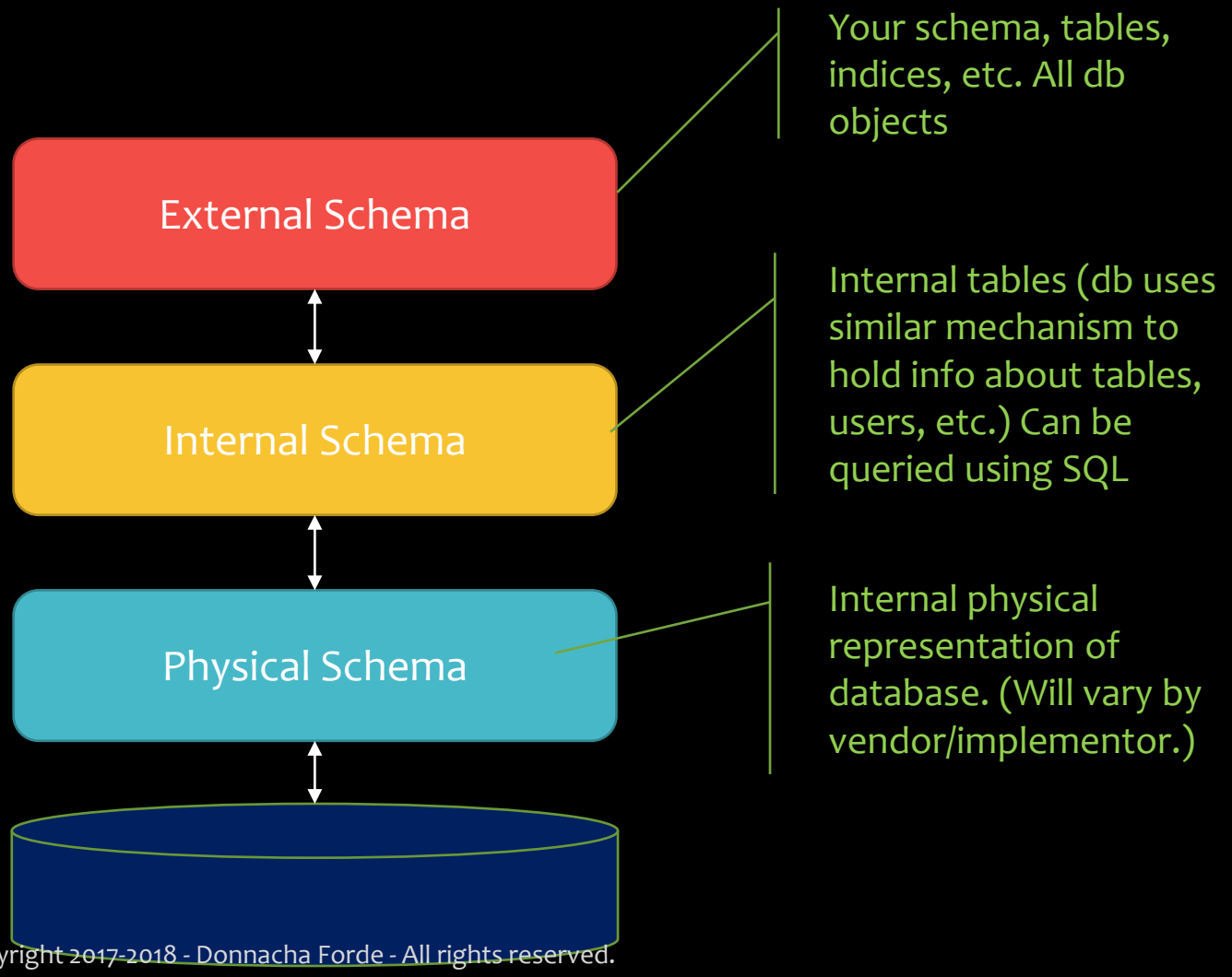
- Google-scale
- Map/Reduce
- Hadoop
- Clustering technology versus enterprise-scale storage arrays

Database Architecture

Tiered Models

Standalone Database Architecture

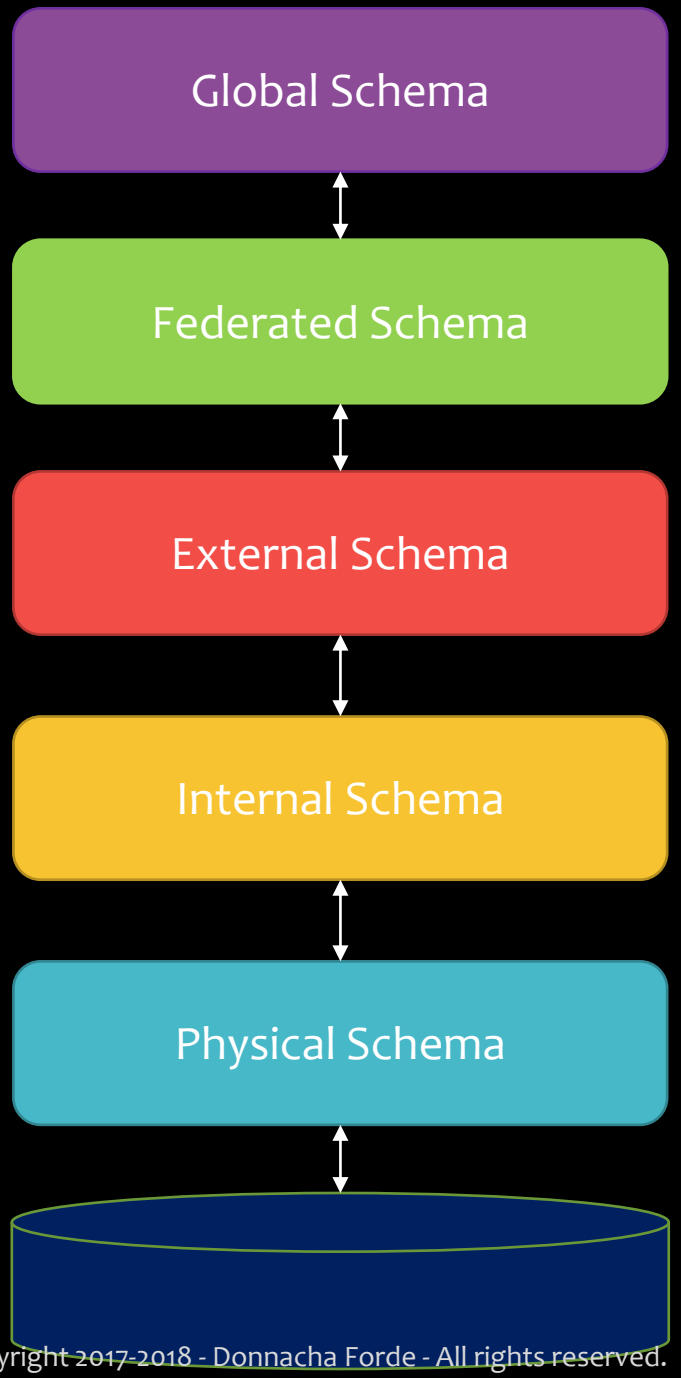
3-layer model



Your schema, tables, indices, etc. All db objects

Internal tables (db uses similar mechanism to hold info about tables, users, etc.) Can be queried using SQL

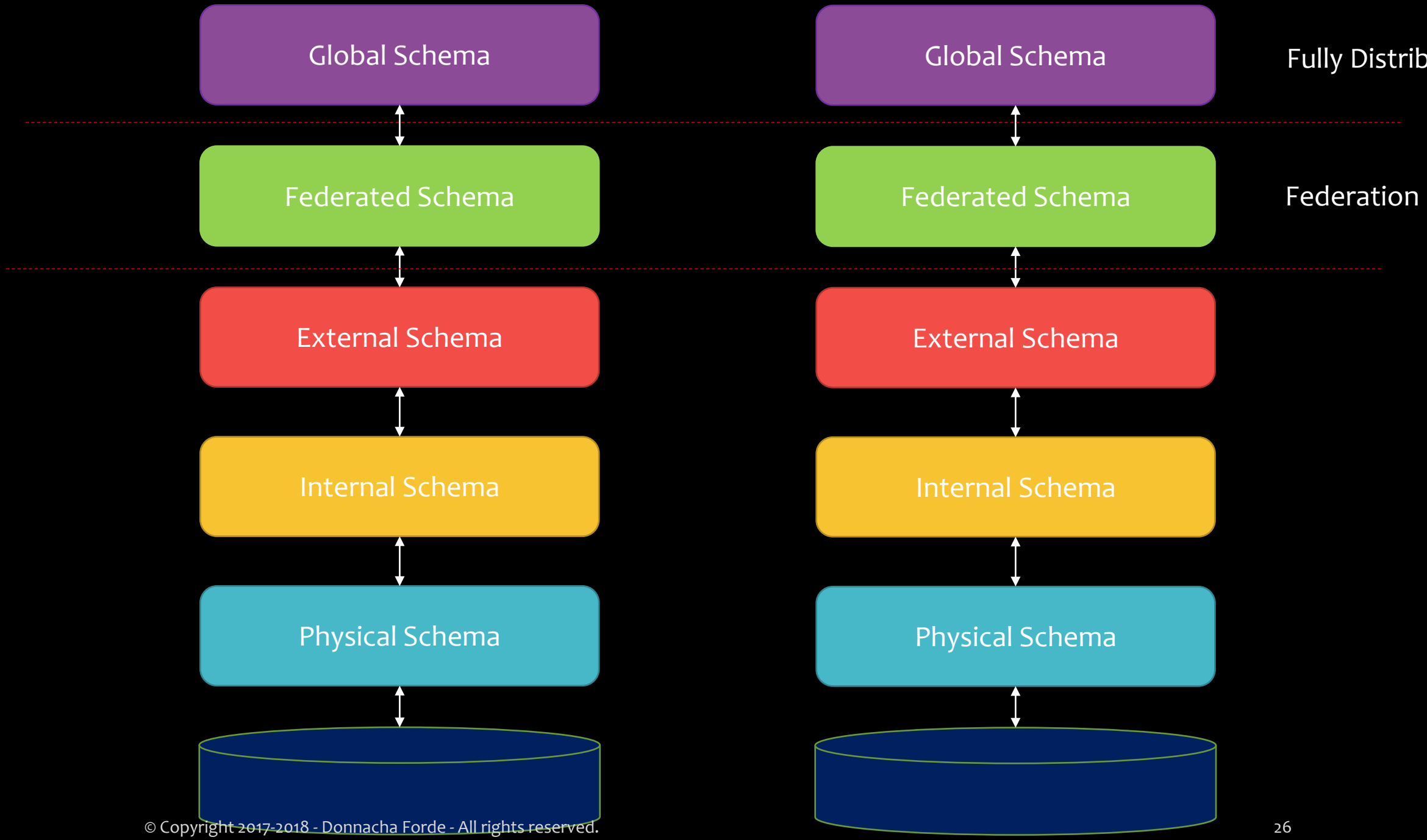
Internal physical representation of database. (Will vary by vendor/implementor.)

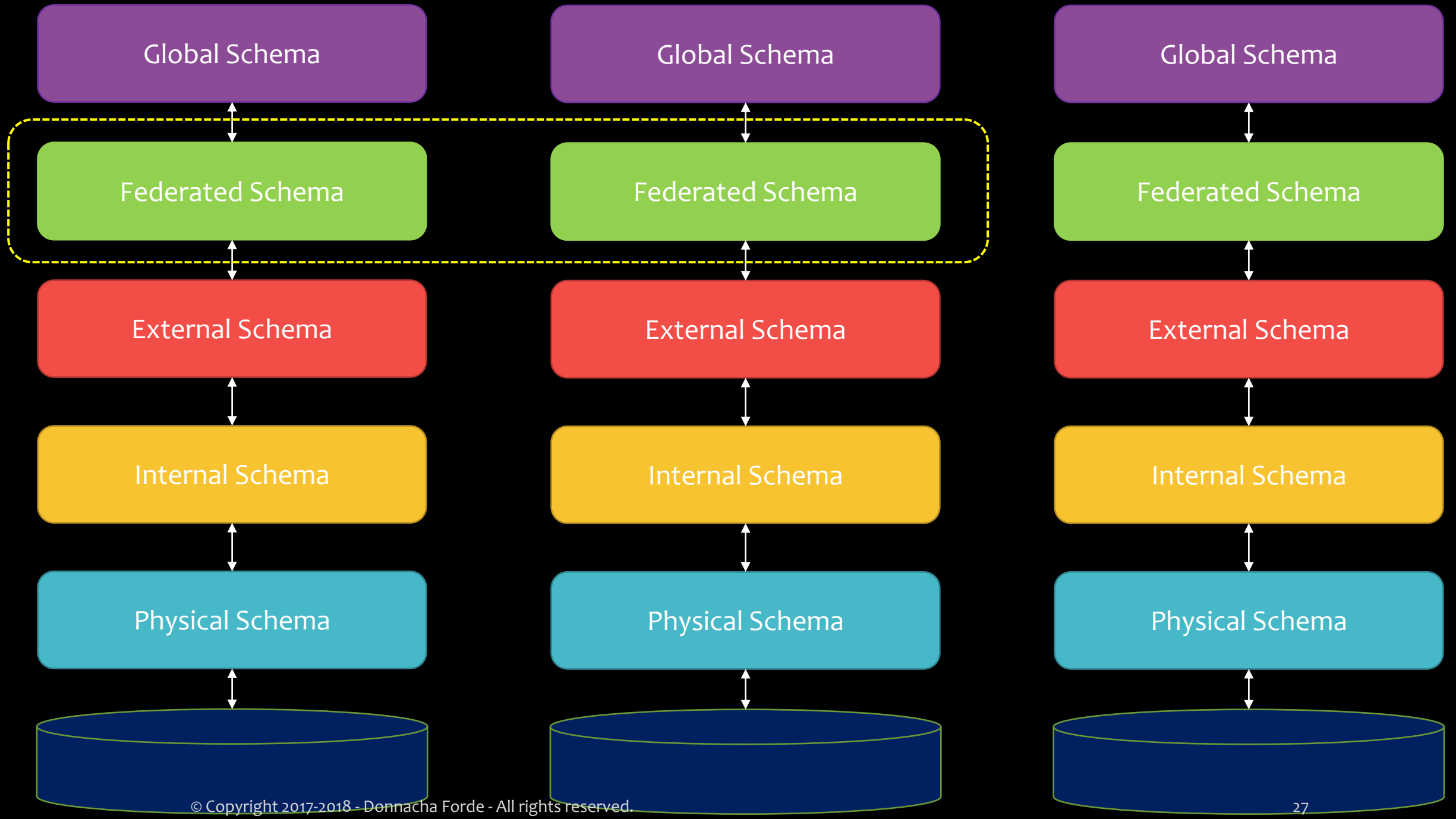


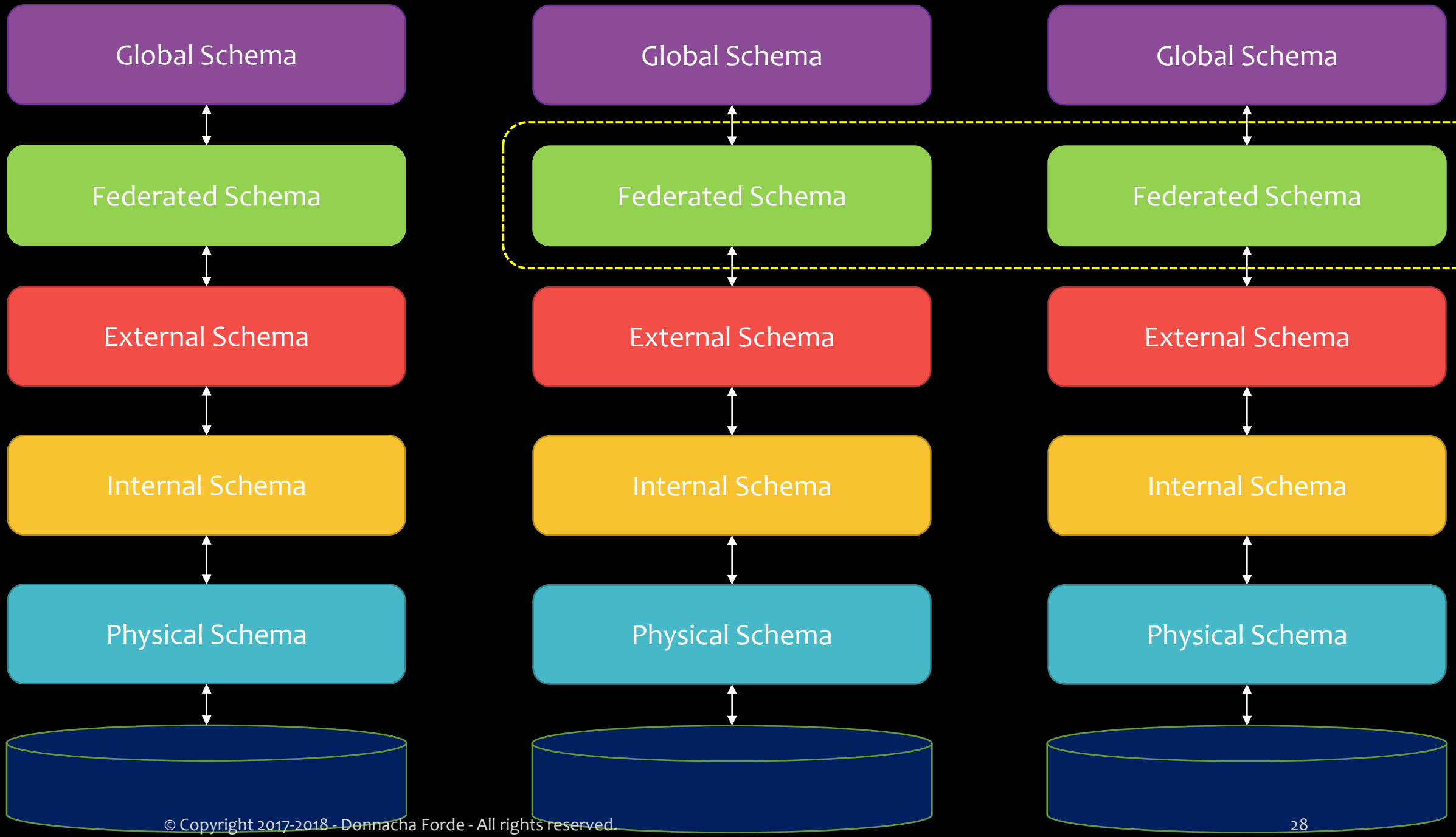
Distributed Database Architecture

5-Layer Model

http://dl.acm.org/citation.cfm?id=96604







Distributed Databases

Various Approaches to collective database technologies

CRUD Analysis→Distributed DB Design

- Matrix Process Function versus Entity
  - Provides CRUD matrix indicating which functions dominate
  - Creates – which functional process create data records
  - Reads – which functional process read data
  - Updates – which functional process update data
  - Delete – which functional process destroy data
- Distributed DB Design – 3 way Matrix of Function/Entity/Location
  - Reveals what data is associated with which parts of the organization
- Conway's Law
  - "the software interface structure of a system will reflect the social boundaries of the organization(s) that produced it, "Logical Schema → Physical Schema

CRUD Matrix Analysis

	View Course	Add Course	Remove Course	Book Place	View Student	Register Student	View Module	Add Module	Delete Module
Student			RU	RU	R	CRUD			R
Course	R	CRU	RUD	R	R	RU	R	RU	RUD
Room	R	R	RU	R	R	R		R	RU
Module	R	RU	RU	RU	R	RU	R	CRU	RUD
Lecturer	R	R	RU	R	R	R		CRUD	RU

CRUD to SQL Mapping

CRUD	SQL
Create	INSERT
Read	SELECT
Update	UPDATE
Delete	DELETE

Limitations to CRUD Analysis

- Conway's Law
  - "the software interface structure of a system will reflect the social boundaries of the organization(s) that produced it "
  - "organizations which design systems ... are constrained to produce designs which are copies of the communication structures of these organizations."

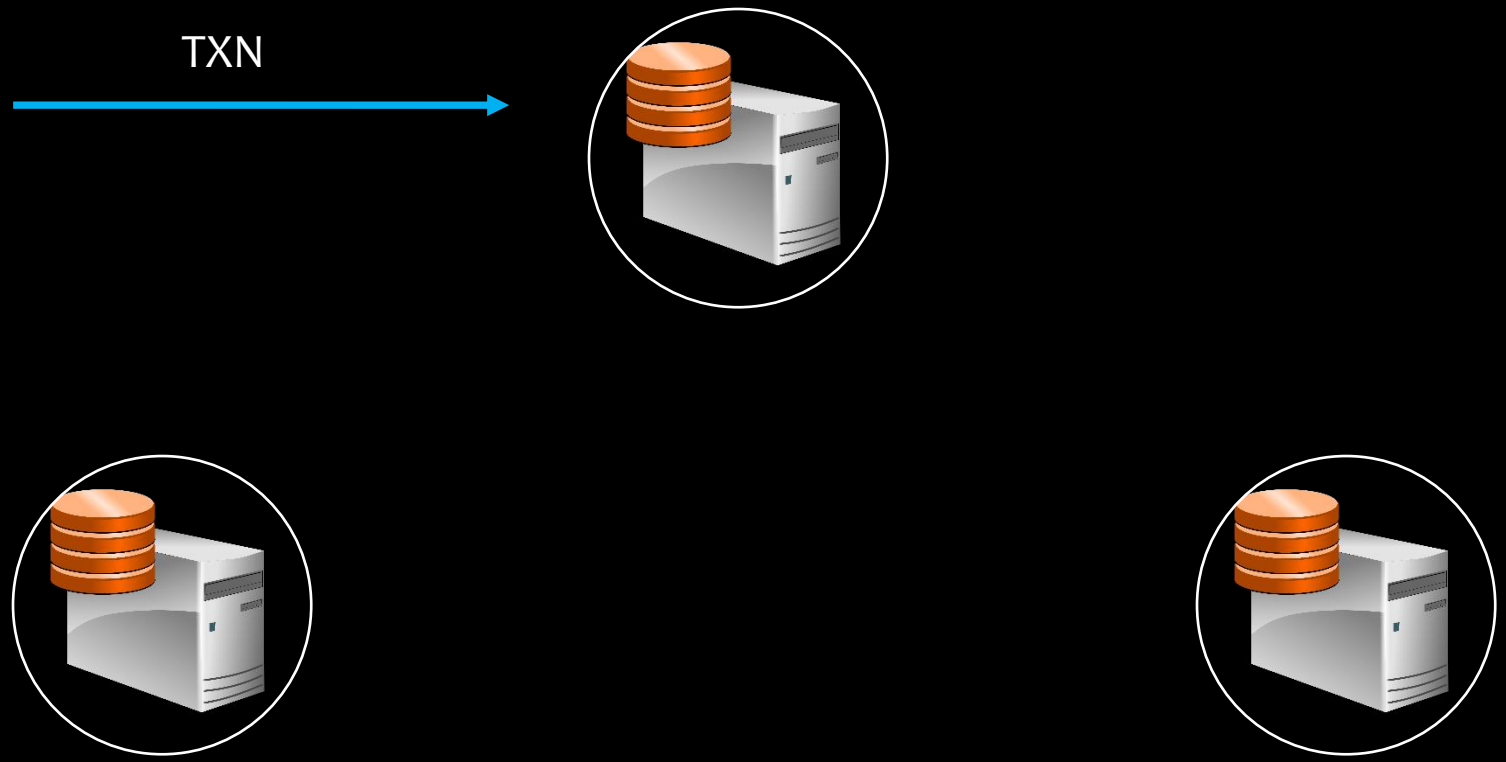
Distributed Transactions

- Simple Transaction
  - Move system from a consistent state, through a change, to a consistent state
- All or nothing
- ACID Criteria
  - Atomic
  - Consistent
  - Isolated/Independent
  - Durable

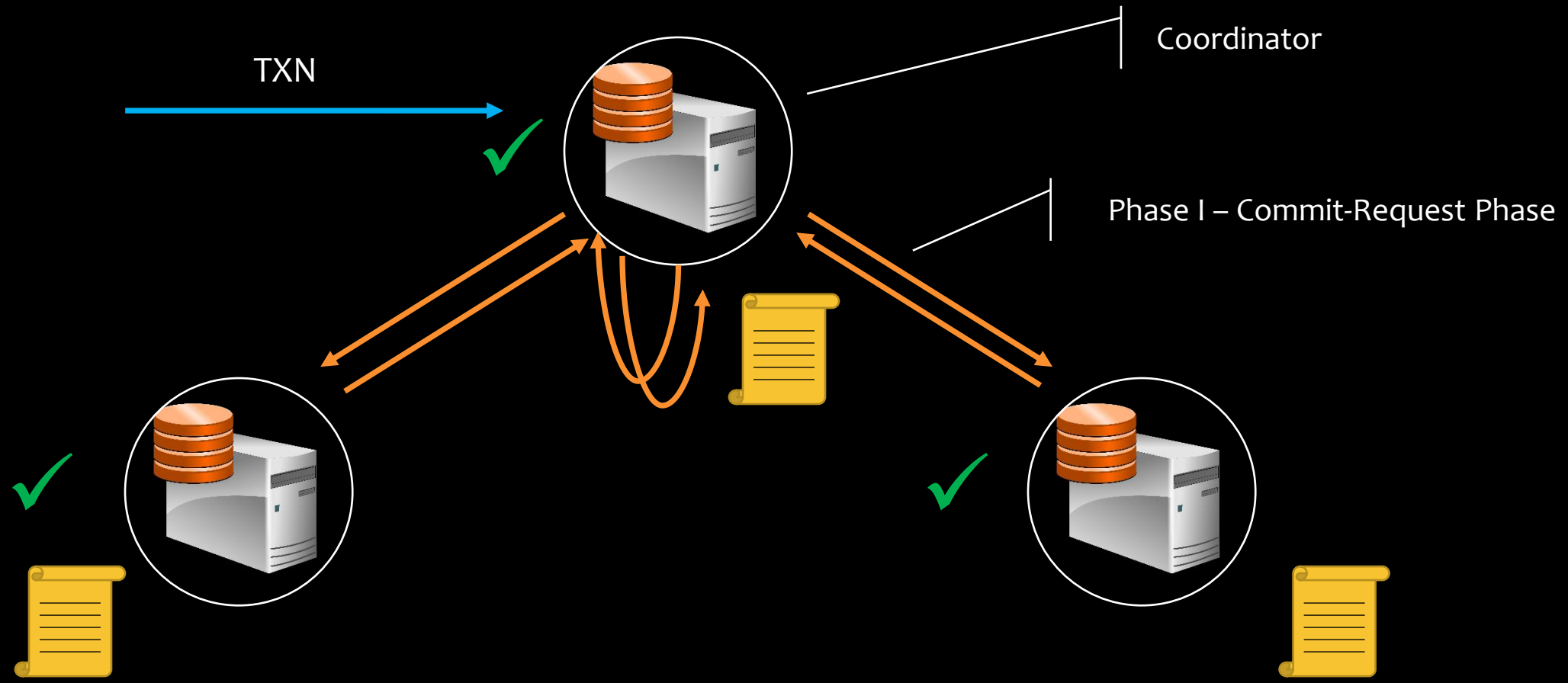
Distributed Transactions

- Distributed Transaction
  - Same semantics except it involves more than one DBMS
  - Move all participants from a consistent state, through a change to a consistent state
- Two-Phase Commit

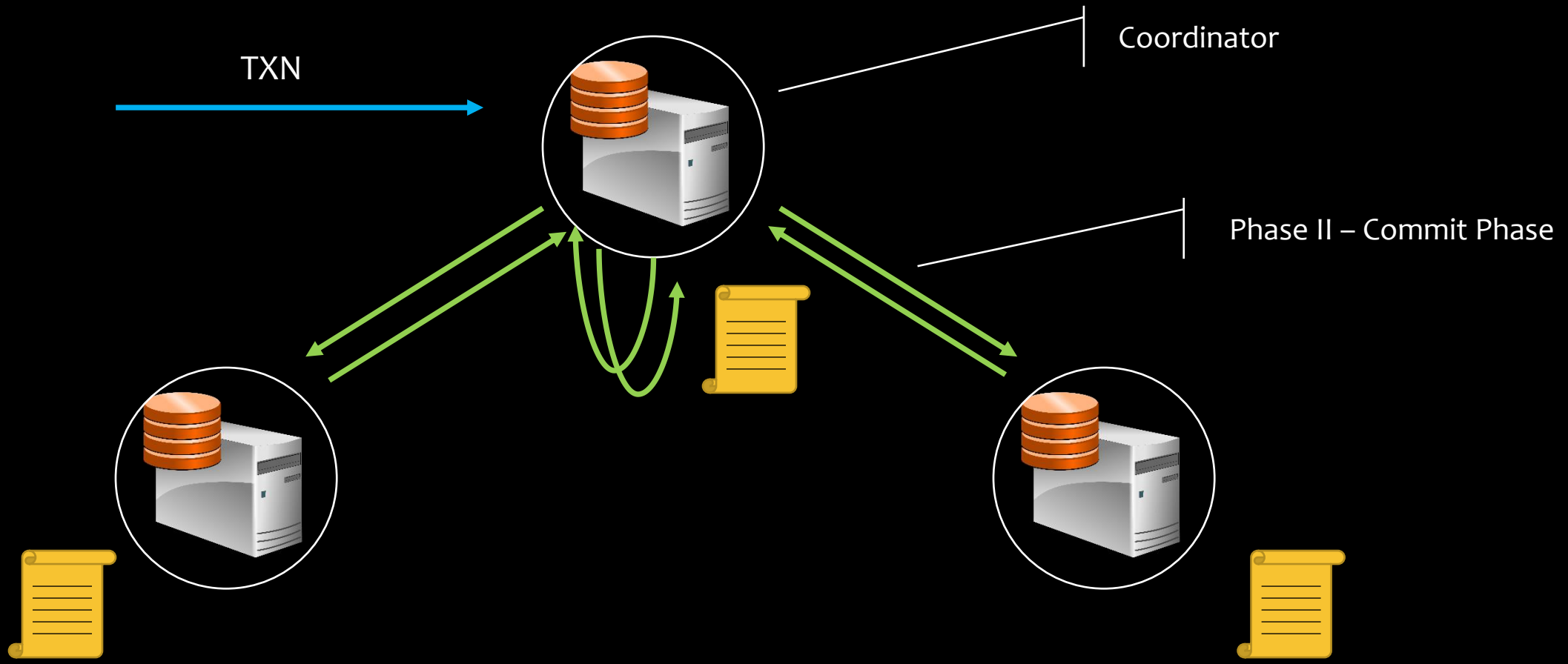
Two Phase Commit – 2PC



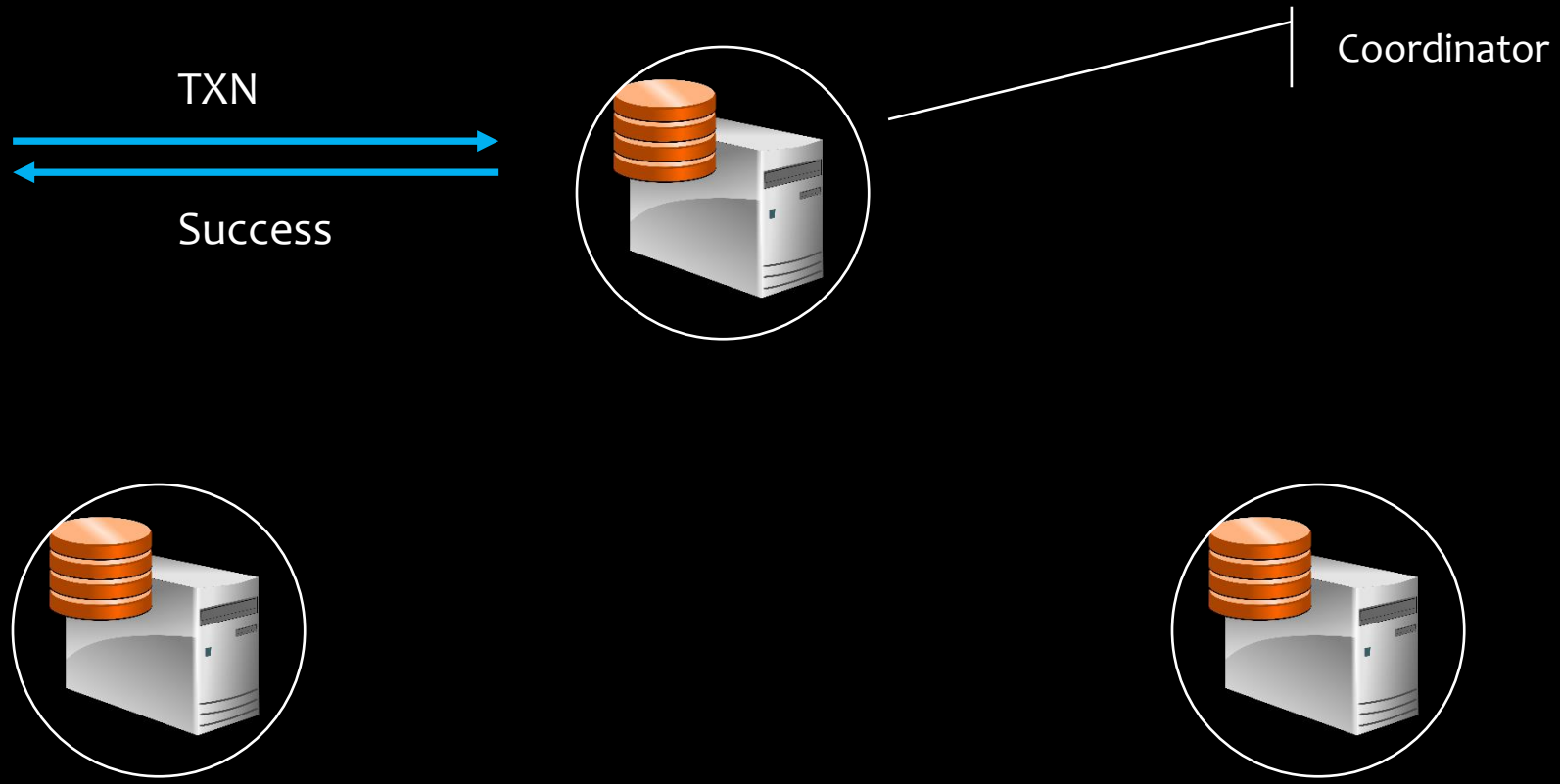
Two Phase Commit – 2PC



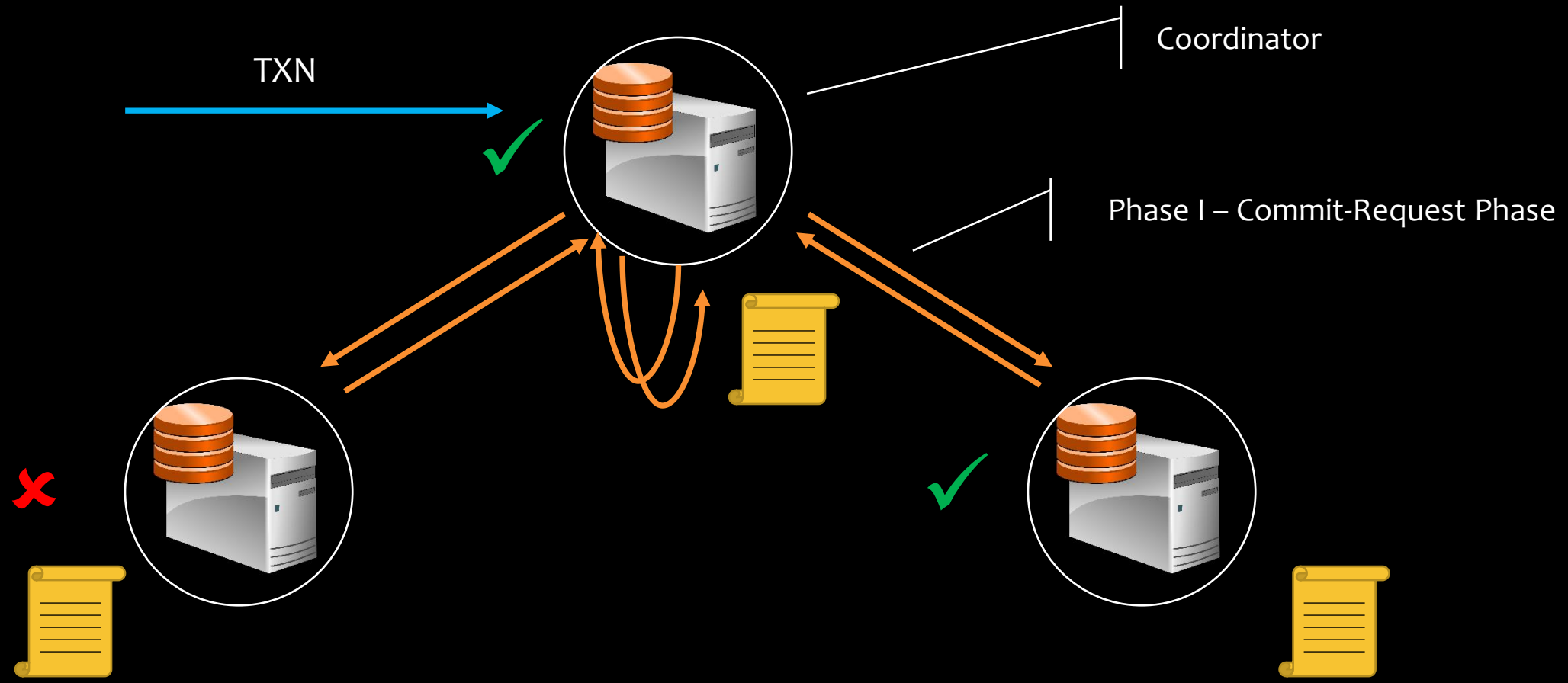
Two Phase Commit – 2PC



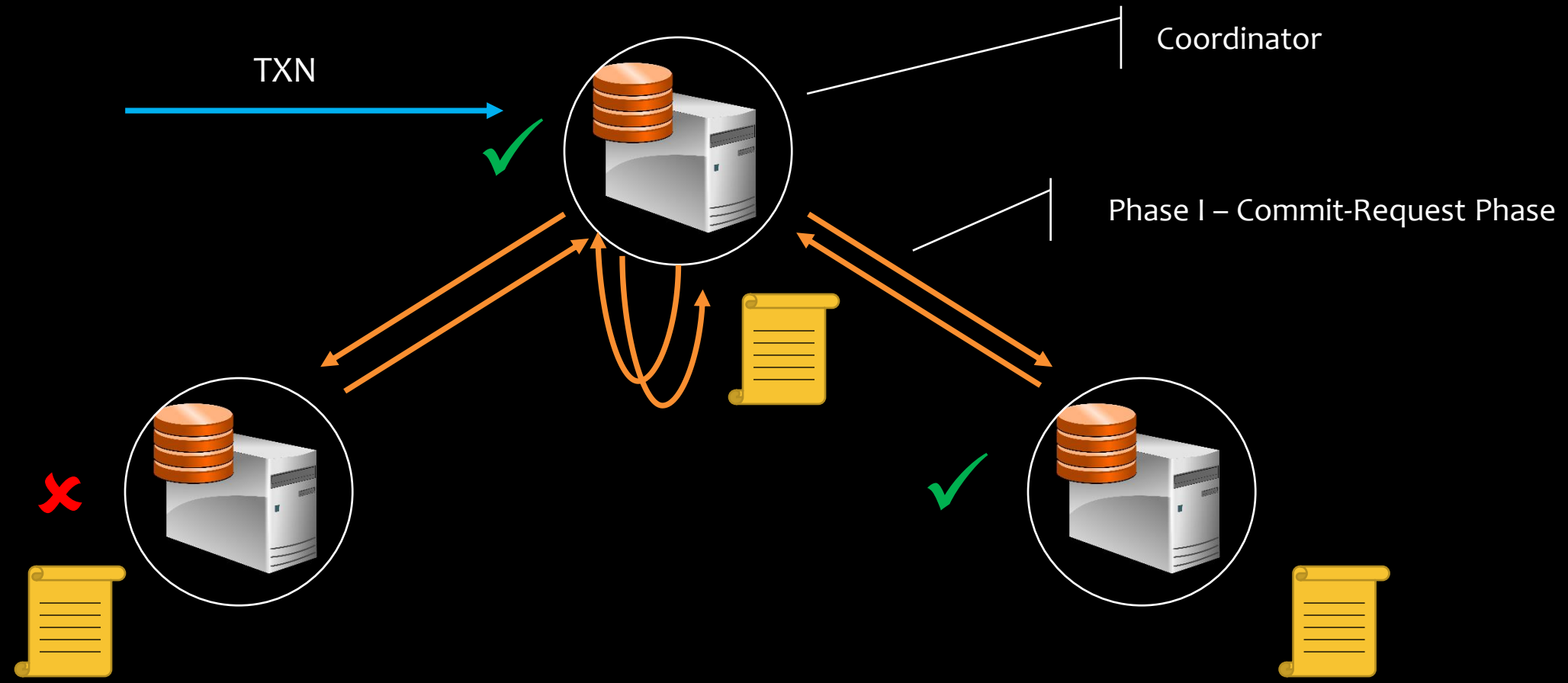
Two Phase Commit – 2PC



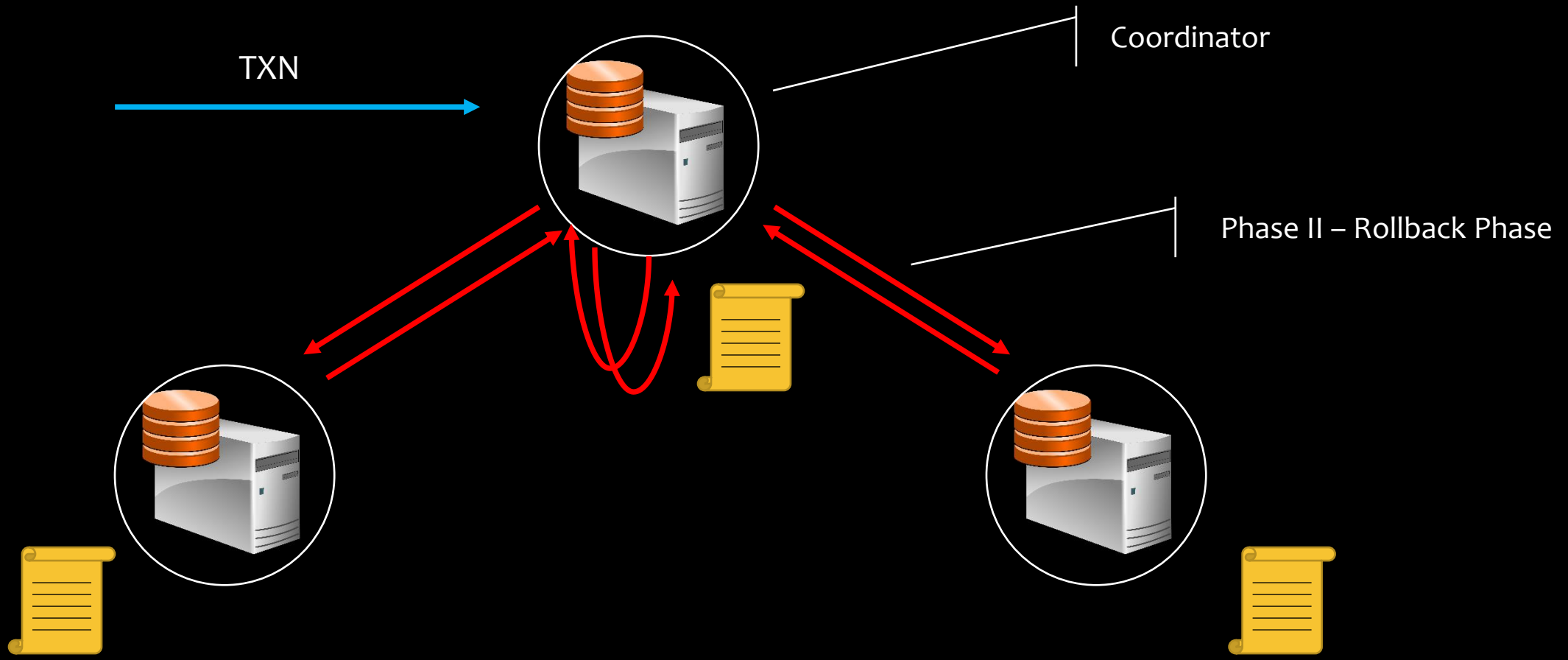
Two Phase Commit – 2PC



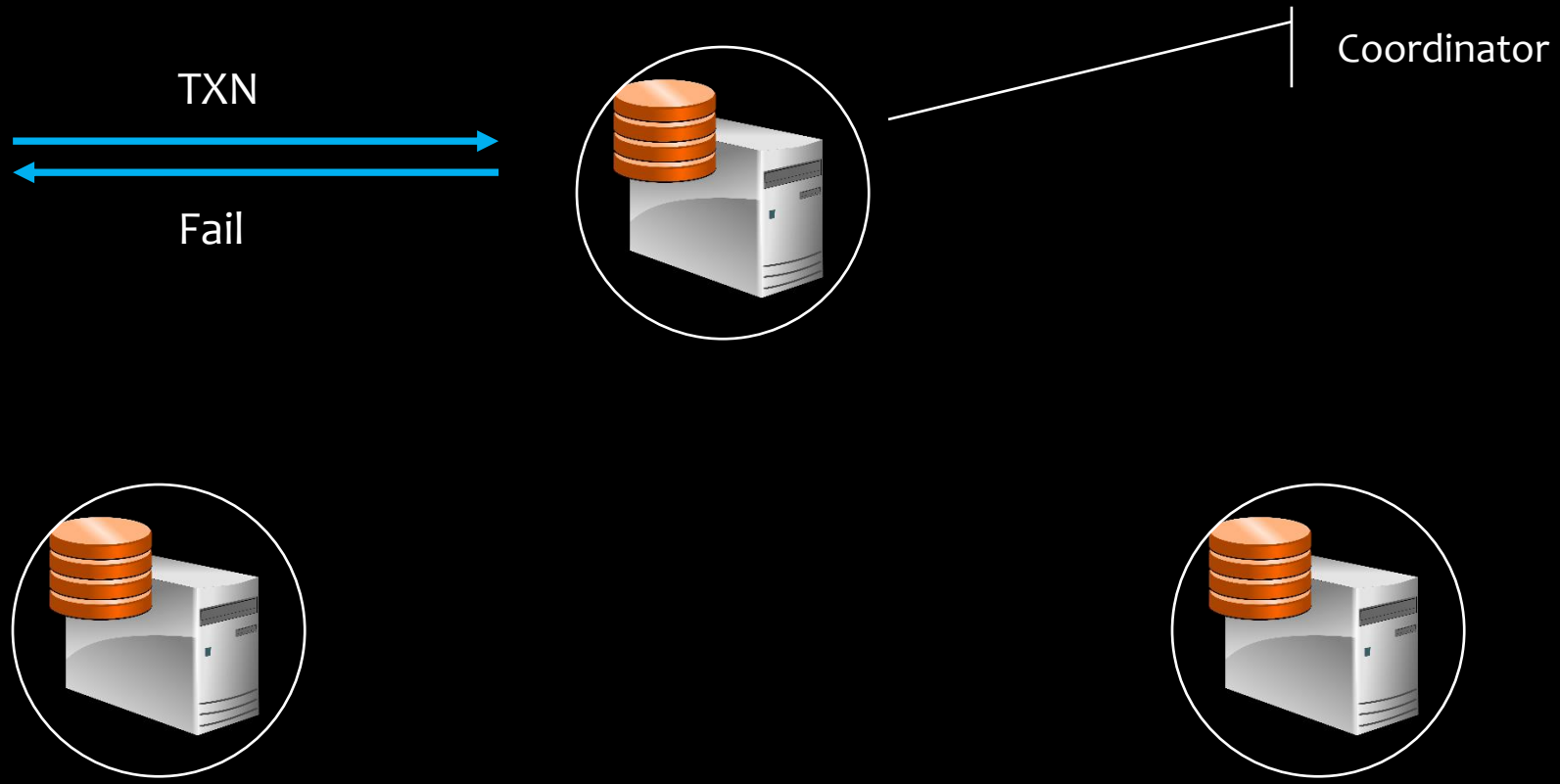
Two Phase Commit – 2PC



Two Phase Commit – 2PC



Two Phase Commit – 2PC



Two Phase Commit - Exercise

Advantages

Disadvantages

Two Phase Commit

Advantages

- Consistency
- Replication → Redundancy

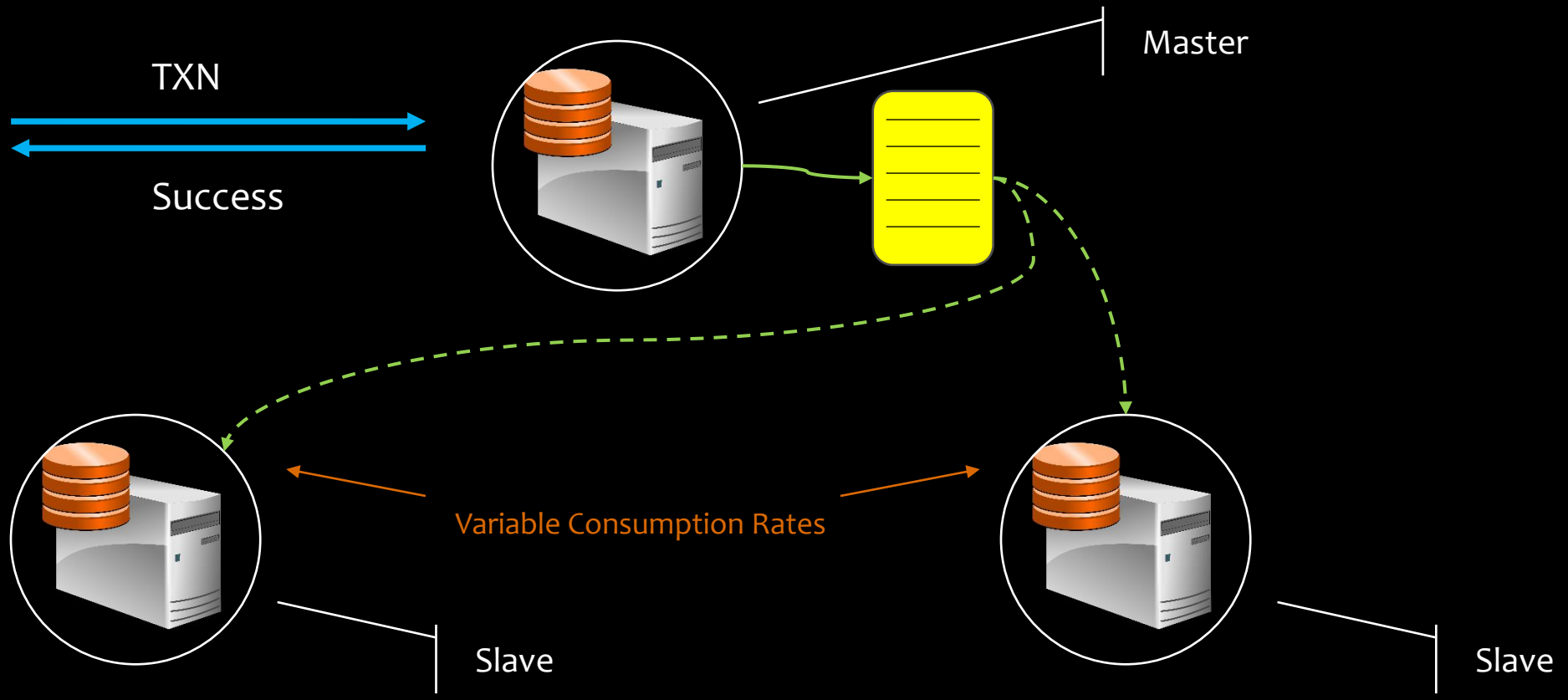
Disadvantages

- Full Availability Required
- Reliable Network Required
- Not highly scalable

Database Replication

- Txns applied to one database are replicated to another
  - ‘Nearly’ up to date
  - Eventual Consistency
- Useful when hard-txn not absolutely necessary
- Useful when one database is predominantly read-only

Database Replication



Database Replication- Exercise

Advantages

Disadvantages

Database Replication- Exercise

Advantages

- Better performant than 2PC
  - More scalable
- Databases are ‘nearly’ up to date
  - Might be ‘good enough’ for many types of applications
  - All databases will be ‘eventually consistent’

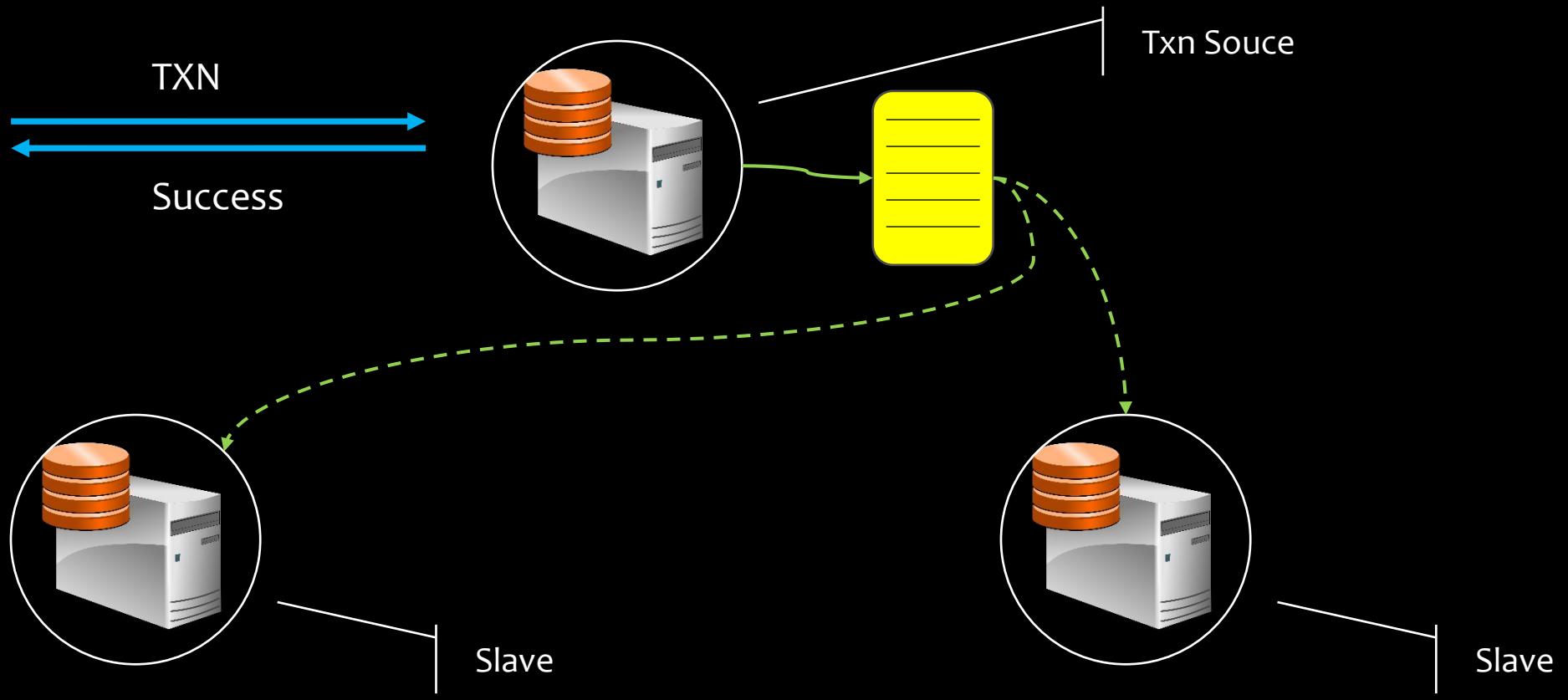
Disadvantages

- Slave database will be ‘slightly’ out of date
  - Could lead to inconsistencies
  - Slaves are slaves
  - Bi-directional is difficult & can lead to inconsistencies
- Scalability limits
  - Wide-Area replication can incur lag, with ‘nearly’ timeframe expanding
- Built-in replication mechanism may be insufficiently flexible
- SAN Replication may be better suited
  - Replication granularity at track level

Transactional Queues

- Suitable when:
  - Participants require transactional-consistency
  - Disconnected – but will eventually connect
- ACID applied through queue
- Txn added to queue **must** be applied

Transactional Queues

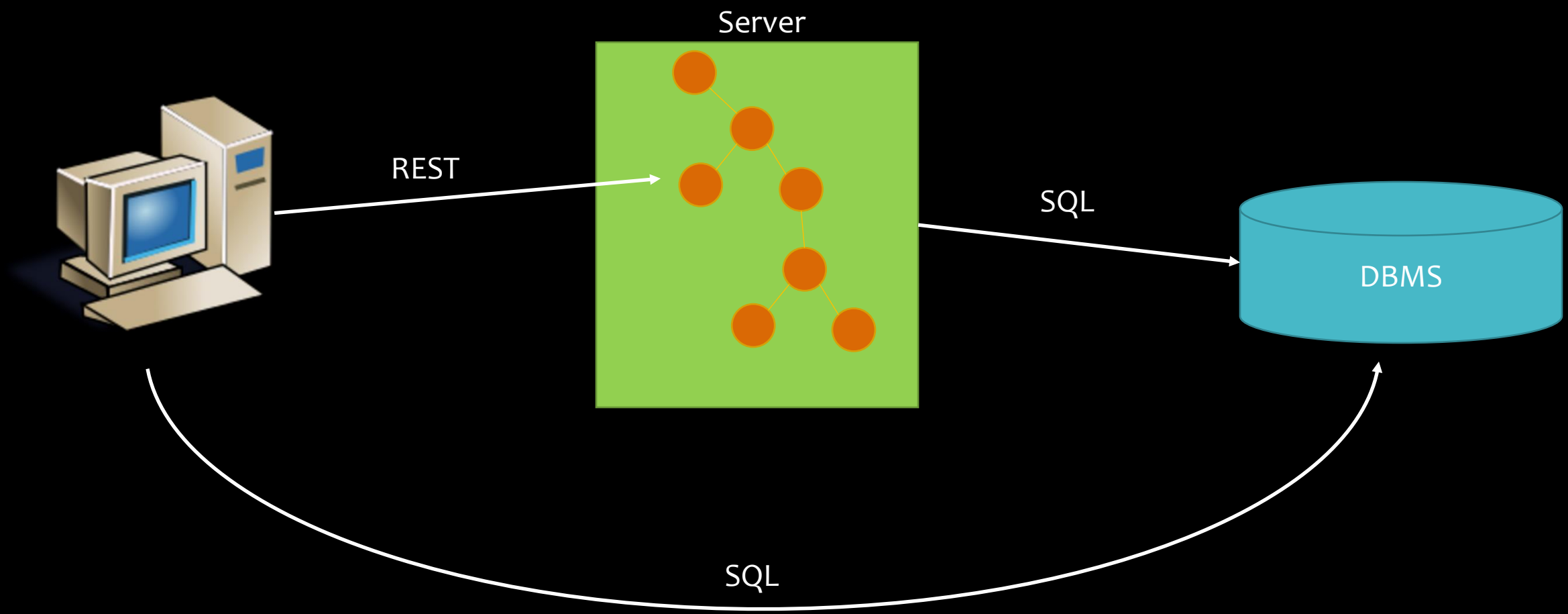


Database Antipatterns

Architecture Issues

Dual Interfaces

- SQL is a form of middleware
- DBMS can be accessed by a myriad of SQL tools & technologies
- Need to determine whether your solution will or won't expose a SQL interface.
- SQL Interface may most likely be internal & private (between your server(s) and the DBMS).



Q&A

Discussion Time

Recommended Reading

- Wiki: SQL, Relational DB
- MySQL Developer Zone: https://dev.mysql.com/doc/

Thank You