DP-900 Notes

ETL vs ELT (Extract Load Transfer)

* ELT is the new school, common in cloud services
* In the ELT pipeline, the transformation occurs in the target data store.
* Storage is so cheap, so we store everything in the original format
* Data Lake support
* ELT is more stream oriented
* Support for unstructured data

Transformation of data (from ELT)

* Simple = mapping 🡪 Data Factory
* Complex = complex analysis 🡪 HDInsight or DataBricks

Batch processing

* processing of blocks of data that have already been stored over a period of time (predefined interval)
* Can output data to a file store or database
* Batch Processing = latency

Vs Streaming

* allows us to process data in real-time as it arrives, and detect conditions within a small period of time from the point of receiving the data.
* It allows us to feed data into analytics tools as soon as they get generated and get instant analytics results
* Very low latency
* Much more expensive

Micro-Batch

* A smaller variant of batch
* smaller accumulations of data

Azure Table Categories

* Fact tables
  + store observations or events
  + in the middle of the star
* Dimension tables
  + describe business entities—the things you model
  + the outer edges of the star
* Integration tables
  + provide a place for integrating or staging data.

Structured Data - Relational DBs

* Are appropriate for scenarios that have a large volume of transactional writes
* Entities going across (rows, records)
* Attributes going down (cols, a particular piece of data)

Semi Structured Data

* Loose or no schema
* JSON/XML (Document type store)
* Parquet (by Apache)
  + Columnar-like storage format
* Partition Key
  + Separate data into logical partitions
  + The goal is even distribution
  + Entities that have the same Partition Key value are stored in the same partition
* Row Key
  + stores string values that uniquely identify entities within each partition
* Primary Key
  + The primary key for an Azure entity consists of the combined Partition Key and Row Key properties

Unstructured – Blob

* Images, video, documents, etc
* Azure Data Lake Storage gen 2 sits on top of blob storage, but adds a hierarchical namespace to it

Key-Value

* Key/Value (Key – Name-Value pairs)
* Inserts and deletes only (no native updates)
* Azure Table Storage uses Key/Value
* Azure Cosmos DB uses Key/value
* Optimized for simple lookups

Document

* Primary entity is a JSON like data structure called a document
* A subset of key/value
* Search queries that a key/value store doesn’t have
* Documents in a document store do not have to have the same structure
* allow for complex data structures, as you can nest documents within other documents

Key/value vs document

* Key/value has no schema defined; document has a schema defined

Columnar

* Stores data as columns (instead of rows)
* Parquet is the format to store to disk
* Lowest latency
* Slower inserts and updates

Time Series Database

* Stores time and some piece of data

Graph

* Data is represented with nodes and structures
* Edges specify the relationship between nodes
  + edges have a direction from node to node
* complex relationships

Normalized vs Denormalized

* Normalized is ideal for transactional workloads or writes – OLTP
* Denormalized is ideal for reads and queries since joins are costly performance wise – OLAP
  + Prejoining tables is one example

Azure Storage Account

* To configure Azure Storage account to support both security at the folder and atomic directory manipulation, you must enable the hierarchical namespace

Azure Blob Storage

* Unstructured data
* Block blobs
  + Max size 4.7TB
  + Best for binary objects that change infrequently
  + Required for Azure Data Lake
* Page blobs
  + Max size 8TB
  + Used for virtual disk storage for Azure virtual machines
* Append blobs
  + Max size 195TB
  + Optimize append operations, ideal for logging

Snapshots – read only version of a blob at a particular point in time

Change feed – provides a read-only record of updates made to a blob

Azure Table Storage

* Key/Value store
* Semi-structured data, so columns in each row can vary
* One to many relationships
* Data is de-normalized, therefore fast queries
* Splits tables into partitions
  + Partition key and row key
* Commonly used for storing TBs of data
* Supports multiple read replicas, but no support for multiple write regions

Index

* An index is an on-disk structure associated with a table or view that speeds retrieval of rows from the table or view
* Speeds up queries, slows down when we perform writes or updates
* Typically, an id of type int (Cosmos db uses this by default)

Clustered index

* An object associated with a table that sorts and stores the data rows in the table based on their key values
* SQL creates this by default on the primary key, so that column is always in order regardless of how the entries were inserted into the table

NonClustered Index

* doesn’t sort the physical data inside the table

File Storage

* File shares in the cloud
* Accessed using SMB 3.0 protocol (Server message block)
* File storage = SMB storage
* Shares can be mounted to a system providing direct access to files in the cloud from anywhere at any time (persistent)
* Supports multiple connections – can be mounted to several systems at one time

Queue storage

* Messages

Storage replication in Azure

* Data in an Azure Storage account is always replicated 3 times in the primary region
  + synchronous- in region
  + asynchronous- different region
* Locally redundant storage (LRS)
  + copies your data synchronously 3 times within a single physical location in the primary region.
  + Least expensive
* Zone-redundant storage (ZRS)
  + copies your data synchronously across 3 Azure availability zones in the primary region.
* **Geo-redundant storage (GRS)**
  + 3 copies in the primary region within a single physical location using LRS, 3 more copies asynchronously in the region pair
* Geo-zone-redundant storage (GZRS)
  + copies your data synchronously across 3 availability zones using ZRS. Then 3 copies asynchronously to a single physical location in the region pair.

Cosmos DB

* Multi-region writes
  + Can be set at creation time or after deployment
  + Cannot use strong consistency with multi-region writes
* Production vs non-Production
  + Affects the Azure Portal experience
* API is configured at the account level

Database

* One or many dbs per account
* Unit of management
* Can Provision throughput (performance, RUs)

Containers

* One or many containers per db
* Need to set the partition key
* Can provision throughput (more common than Database level)
  + The minimum throughput for containers is 400 RU/s
* Once a container has been created you cannot change properties

Item

* Individual record
* Includes system defined properties
* Cannot mix APIs in a single Cosmos DB account
* Cannot change API once account has been created

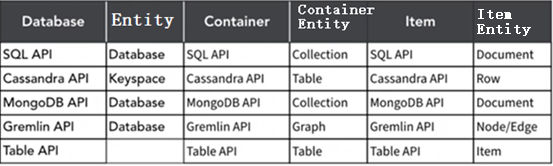
Cosmos DB APIs – set when you create the account

* SQL API
  + Recommended #1
  + JSON Documents
* Gremlin API
  + Recommended #2 if you have complex relationships
  + Graph data
  + Can query using an SQL like language to produce a JSON document
  + Nodes/Edges
  + A node is some Object
  + An Edge is the relationships between Objects
    - Edge can have direction and data properties
  + Many to many relationships
* MongoDB API
  + JSON Documents
  + Programmatic interface
* Table API
  + Supports multiple read replicas and supports multiple write regions
  + Key/value data
  + Advantage over Azure Table Storage - supports a multi-master model
  + Query examples -

SELECT \*

FROM Products p WHERE p.id = “1”

* Cassandra API
  + Column based storage



Request Unit (RU)

* Encapsulation of many of the resources needed for the db into a single unit
  + Cpu, memory, IOPS (input/output operations per second)
* 1 RU == 1kb item read operation from a Cosmos db container
* Provisioned or auto-scale
  + Assigned at the db level, or container level

Cosmos DB Aggregate functions

* COUNT ( <fields\_to\_count> )
* SUM ( <numeric\_fields> )
* AVG( <numeric\_fields> )
* MAX ( <numeric\_fields> )
* MIN ( <numeric\_fields> )

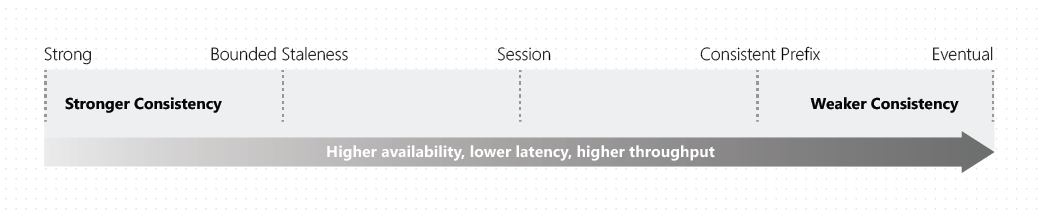
Cosmos Data Consistency Levels (config’d in Default Consistency in the account level)

* Strong
  + Also called immediate consistency
  + guarantees that reads get the most recent version of an item
  + only one that offers a linearizability guarantee (in order)
  + data is consistent for all observers of the entity
  + data has to be locked during the period of update or replication process to ensure that no other processes are updating the same data
  + reduced availability (lag)
  + requires a relational database
* Bounded staleness
  + guarantees that a read has a max lag - either number of versions (K) or time (T)
  + Maximum Lag (Operations) or Maximum Lag (time)
  + apps that expect low write latencies but require total global order guarantee
  + great for apps featuring group collaboration and sharing, stock ticker, publish-subscribe/queueing, etc;
* Session
  + The default consistency
  + Most balanced
  + guarantees that a client session will read its own writes in a session
  + best fits apps in which write operations occur in the context of a user session
* Consistent prefix
  + guarantees that updates are returned in order, but no guarantee you can read all the data
  + Example – if you write “A, B, C” you can read “A”, “A, B” or “A, B, C” but never “A,

C” or “B, A, C.”

guarantees that reads never see out-of-order writes

* Eventual
  + provides no guarantee for order of data
  + the only one with out of order reads
  + high availability and better performance (lowest latency)
  + different data (inconsistent) available in different regions
  + Use this consistency level if the order of the data is not essential for your app, such as non-threaded comments
  + When data is written, it will eventually be in sync with other instances
  + all reads of the entity will eventually return the last updated value



Online analytical processing (OLAP)

* Large volumes
* Capturing raw data
* SQL statement - SELECT
* Consists of software tools that are used for data analysis
* Offline storage of data
* Typical usage - Data warehousing solution to run analytics and produce reports
* organizes large business databases and supports complex analysis.
* It can be used to perform complex analytical queries without negatively affecting transactional systems
* Used by data analysts
* Azure solutions
  + Azure Synapse

Online Transaction Processing (OLTP)

* Higher volume of small day to day transactions
* Fast access of data
* SQL Statement – INSERT, UPDATE, DELETE
* Does support simple queries
* Typical usage – online retailer writing data to database frequently
* record business interactions as they occur in the day-to-day operation of the organization, and support querying of this data to make inferences
* used by clerks, cashiers
* heavy writes and moderate reads, schema on write, normalized data (3rd Normal Form)
* Example - an e-commerce application that reads and writes data to an Azure SQL database
* Azure Solutions
  + SQL Server
  + Azure Database – Postgres, MySQL, MariaDB (all open sourced, all managed)

SQL Server

* Can run in different ways
  + IaaS – install SQL Server onto VM
  + PaaS – Azure SQL DB
* Relational DBs
  + Creation of tables with defined columns
  + Strongly typed schema
  + Primary key – must be unique for every record in the table
  + Foreign keys to reference keys in other tables (see Star schema)

Azure SQL family

* SQL server on Azure Virtual Machines
  + When you need to Lift and shift
  + OS level access
  + IaaS
* SQL Managed Instance
  + Ideal for migrations to the cloud
  + PaaS
  + Multiple db’s on a single managed instance
* Azure SQL Database
  + Fully managed SQL dbs
  + PaaS
  + Protected by a server-level firewall by default
  + An internal firewall the resides on the db server
  + Includes a fully managed backup service
  + Has built in high availability
  + Auto scaling
  + Need to create Azure SQL db server first (logical server) which is a central administration point
  + No patching, always running the latest and most secure version
  + Optional Azure Advanced Threat Protection (ATP) – expensive as shit
  + Supports OLTP
* Azure db PaaS options are designed to be always on unlike IaaS virtual machines which can be stopped

Open Source relational DBs in Azure

* MySQL
* MariaDB
  + A fork of MySQL
* PostgreSQL
  + The most advanced relational db

Network security for SQL Databases

* Allow access to Azure services
* Firewall rules
* Virtual network rules
* Private link
* SQL standards are maintained by ANSI and ISO

SQL statement types

1. DML – Data Manipulation Language
   1. Manipulating data inside the db
   2. SELECT, INSERT, UPDATE, DELETE
      1. SELECT, FROM, WHERE, GROUP BY, ORDER BY
      2. GROUP BY is often used with aggregate functions (COUNT(), MAX(), MIN(), SUM(), AVG()) to group the result-set by one or more columns.
2. DDL – Data Definition Language
   1. Creating or modifying the db or db objects
   2. CREATE, ALTER, DROP, RENAME, COMMENT, TRUNCATE
3. DCL – Data control language
   1. GRANT, REVOKE

Views in SQL

* A view is a virtual table whose contents are defined by a query
* CREATE VIEW [ <name of view> ] AS [ <query> ]

Cosmos DB Key/Value data stores

* associates each data value with a unique key.
* only support simple query, insert, and delete operations.
* Optimized for simple lookups

Datastores in Azure

* Can be unstructured or semi-structured data
  + Data Lake
  + Data Warehouse
  + Database
  + Files
  + Data requested from API endpoint from a webapp

Transparent data encryption (TDE) for SQL

* encrypting the database at rest
* real-time I/O encryption and decryption of the data at the page level

Connectivity & Firewalls

* Azure SQL DB connects over port 1433
* Opening to a range of 0.0.0.0 to 255.255.255.255 opens the machine to every ip on the planet

3 types of server connection policy

1. Redirect (recommended)
   1. Clients establish connections directly to the node hosting the database, leading to reduced latency and improved throughput
   2. Can use Service tags which are a group of ip addresses
2. Proxy
   1. all connections are proxied via the Azure SQL Database gateways, leading to increased latency and reduced throughput.
3. Default
   1. Established by default unless you explicitly alter the connection policy to either Proxy or Redirect
   2. The default policy is Redirect for all client connections originating inside of Azure (for example, from an Azure Virtual Machine); And Proxy for all client connections originating outside (for example, connections from your local workstation)

Azure Services RBAC

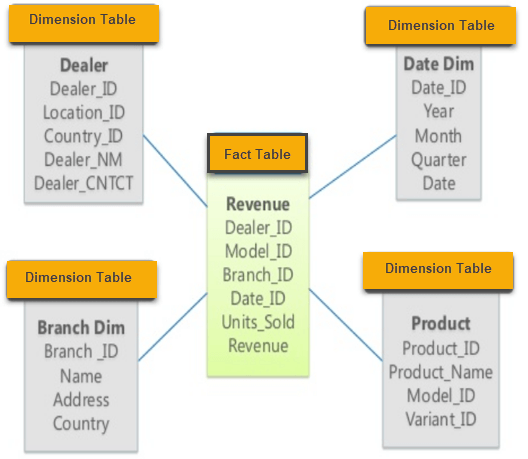
* SQL DB Contributor
  + Manage SQL Db’s, but not access them
* SQL Managed Instance Contributor
  + Manage SQL managed Instances
* SQL Security manager
  + Manage security policies of SQL servers and db’s
  + No access to SQL servers
* SQL Server contributor
  + Manage SQL servers and dbs
  + Not access to SQL servers

Azure db – read only replicas

* Read replicas help improve performance and scale of read intensive workloads such as Power BI and Synapse Analytics
* With PostgreSQL you can create up to 5 replicas of the master server

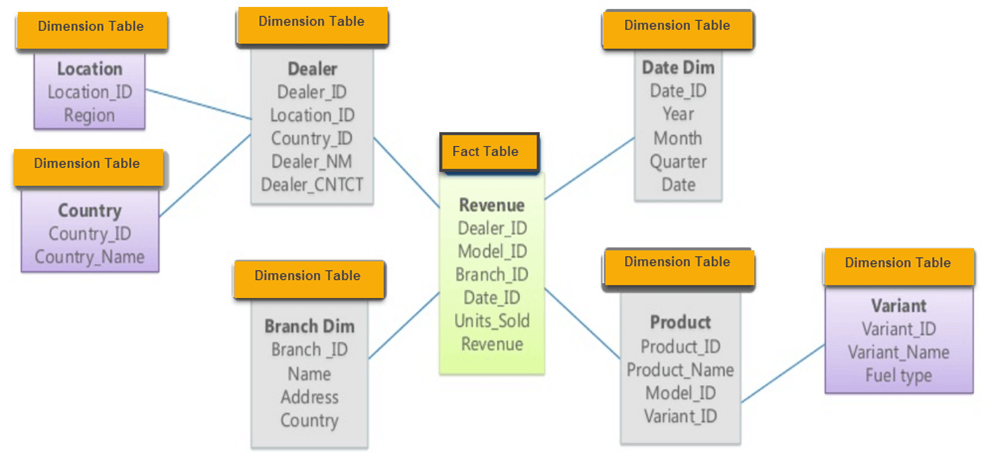
Star schema

* a mature modeling approach widely adopted by relational data warehouses
* requires modelers to classify their model tables as either dimension or fact



Snowflake schema

* The goal 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.
* Multi-dimensional structure
* Dimension tables link to other dimension tables

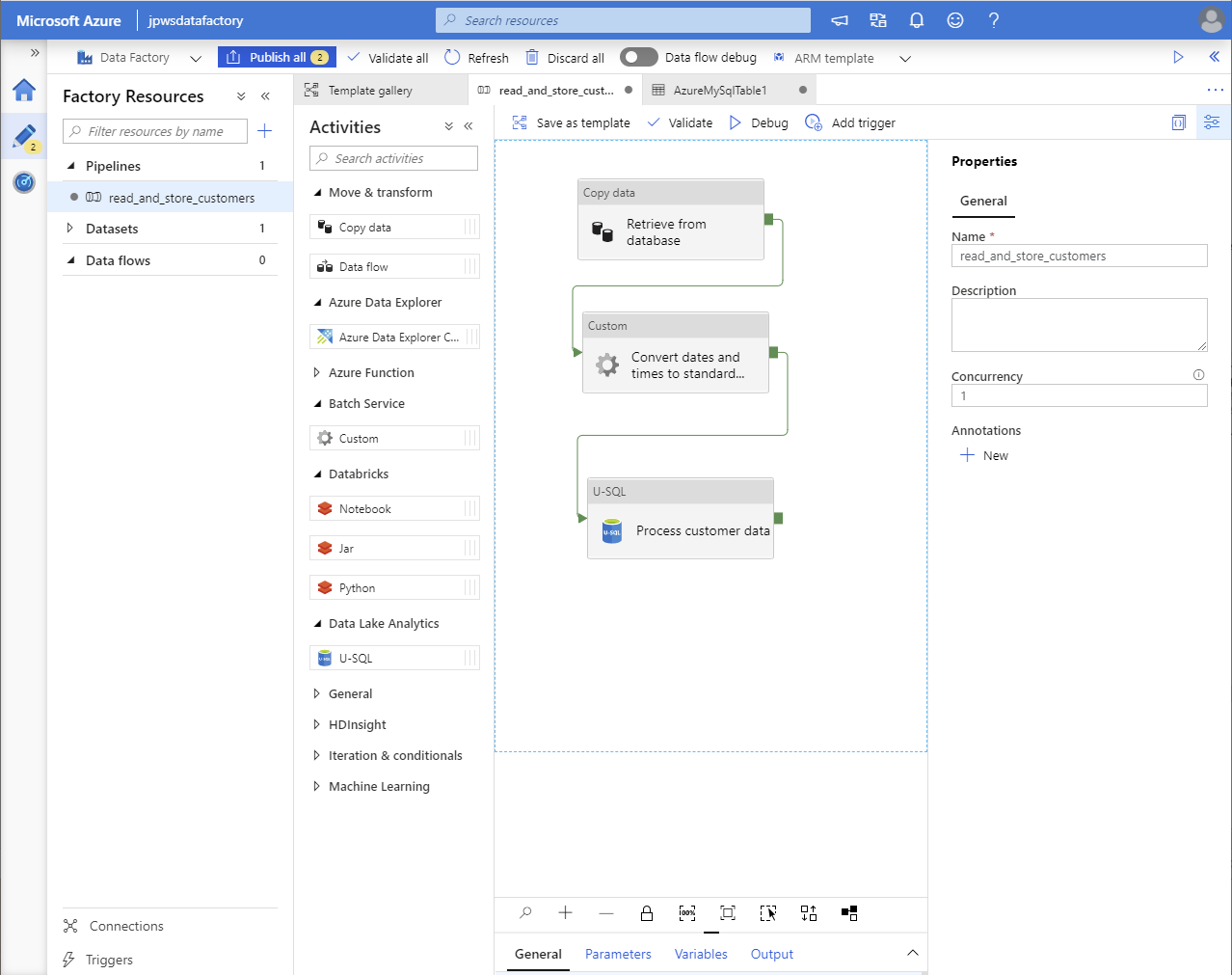


Galaxy schema

* Multiple stars
* Multiple fact tables in one schema

Azure Data Factory

* Pipeline builder - ie looks like MS Visio
* Orchestration of ELT or ETL
* data integration service
* retrieve data from one or more data sources, and convert it into a format that you process
* data can be static and/or streaming
* Get data from source and get it to the sink using a pipeline of operations
* A cloud based data injection service that allows you to orchestrate and automate data movement and data transformation
* Build ELT Pipelines visually without writing any code via web interface
* moves data from a data source to a destination via linked services
* Can Output data to Parquet format
* Orchestrates data integration workflows
* Can use control flow to orchestrate pipeline activities that depend on the output of other pipeline activities
* Dataset
  + A representation of data structures within data stores
  + represents the data that you want to ingest (input) or store (output)
  + must be created from paths in Azure datastores or public web URLs, for the data to be accessible by Azure Machine Learning.
* Linked service
  + The information used to connect to external resource
  + Similar to connection strings, which define the connection information needed for Data Factory to connect to external resources
* Pipeline
  + A logical grouping of activities that performs a unit of work and can be scheduled
  + A pipeline can execute other pipelines
  + A step within a pipeline is an activity
  + Use triggers to initiate pipelines
* Control flow
  + Used to orchestrate pipeline activities that depend on the output of other pipelines
* Integration runtime
  + provides the compute environment for activities
* Data movement activities
  + Copy
* Data transformation activities
  + Mapping data flow
* Control flow activities
  + Until (Activity) – a Do-Until loop
  + Lookup
  + Switch
* PolyBase
  + File based
  + Useful when you need to use Transact-SQL to query files in Azure Data Lake Storage from an Azure Synapse Analytics data warehouse
  + allows the data to stay in its original location and format.
  + a feature of SQL Server and Azure Synapse Analytics
  + makes these external data sources appear like tables in a SQL database.



Data Lake

* Datastore
* Data repository for large quantities of raw data in semi-structured or unstructured format
* Staging point of injected data before it’s converted
* Sits on top of blob storage (and extension of Azure Blob storage)
* Generally use object (blobs) or files as its storage medium
* Pool of data with an unknown purpose or insight
* Azure Storage account must be Premium tier, Block Blob and have hierarchy namespace enabled (under the advanced tab)
* data is typically loaded into Azure Data Lake Storage before being processed into a structure
* an extension of Azure Storage Account
* that enables efficient analysis in Azure Synapse Analytics
* Supports role-based access control (RBAC) and Access Control Lists (ACL) at the file and folder level
* Can store data in Parquet format
* Incur costs for ingesting data across Azure regions
* Azure resource group

|

Azure storage account

|

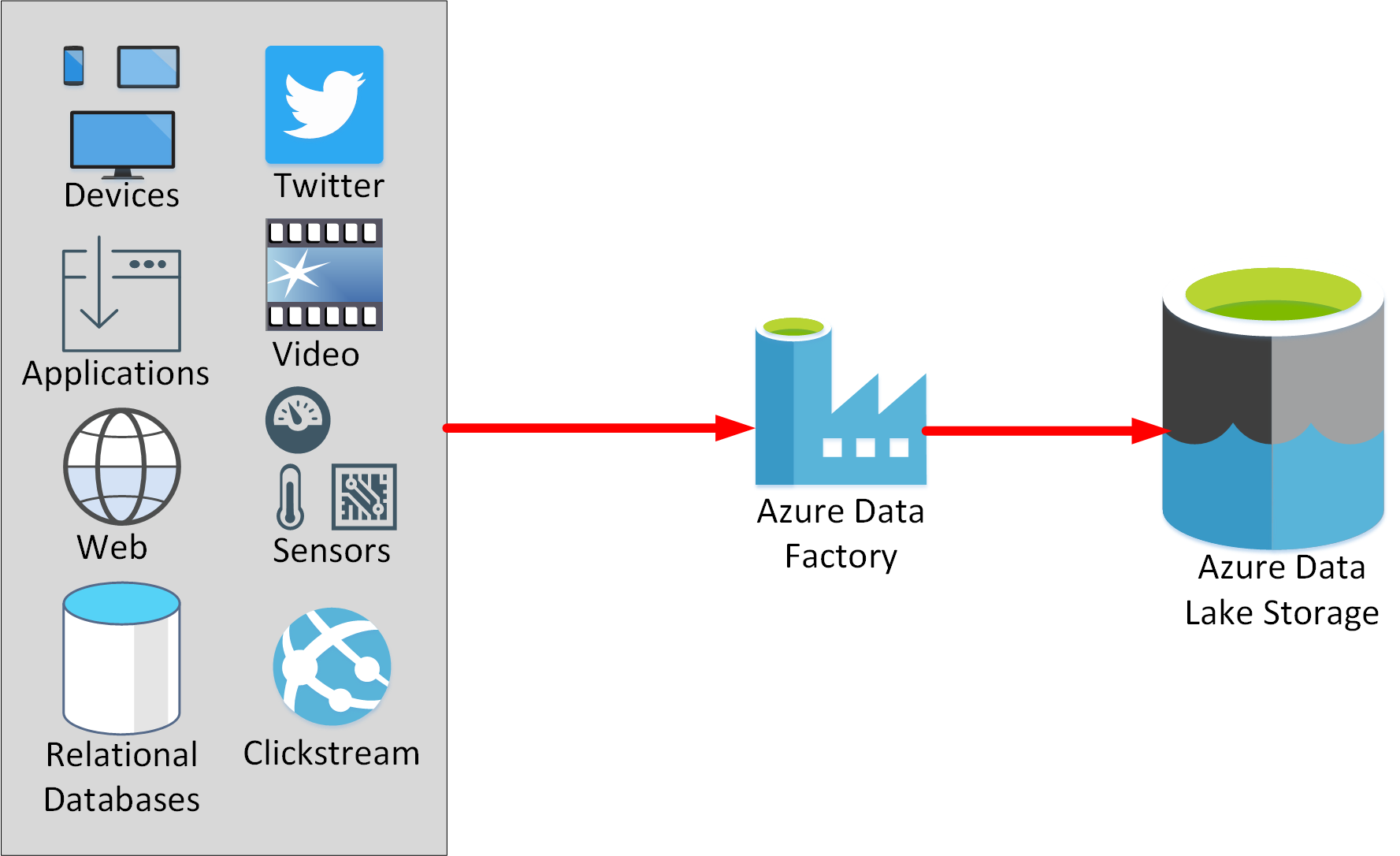
Container

|

Folders

|

Files



Data Warehouse vs Data Lake

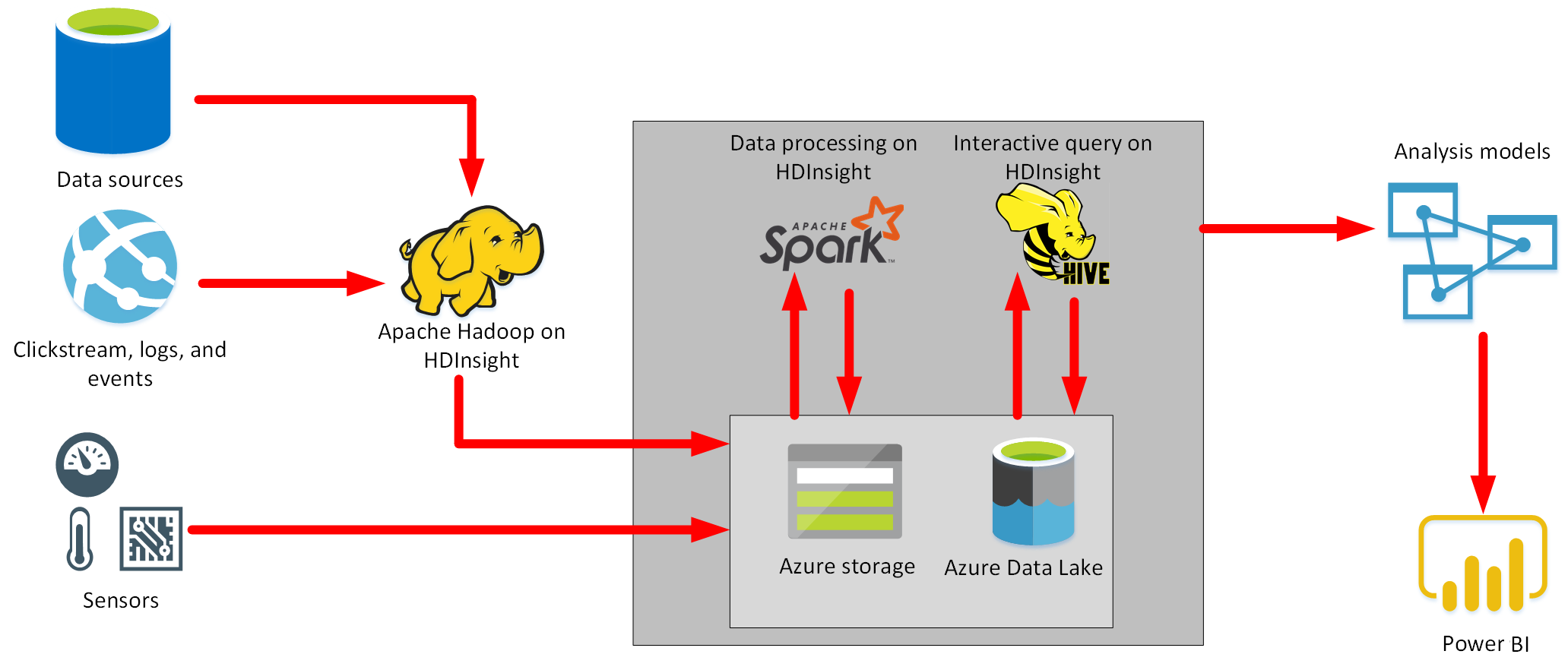
* Both are data stores
* A data lake holds raw data, but a data warehouse holds structured data

Azure Databricks

* 3rd party Apache Spark based platform
* can consume data from Azure SQL database, Event Hubs and Cosmos DB
* data cleaning and preparation
* graphical user interface where you can define and test your processing step by step, before submitting it as a set of batch tasks
* Integration w/ Azure services (Azure data platform and Power BI)
* Big data collaboration platform (PaaS)
* data processing engine that can consume and process large amounts of data very quickly
* Similar to HDInsight except the clusters we create are based on Apache Spark alone
* Help with data transformation at large scale
* Run queries on your data lake

Azure HDInsight

* Big data processing service
* Flexible multi-purpose big data platform (PaaS)
* Service for open-source analytics such as Apache Spark or Apache Hadoop
* Provides open-source big data clusters
* Similar to Azure Synapse Analytics
* can also be used instead of or in conjunction with Synapse Analytics

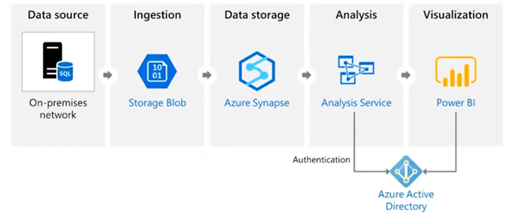


Azure Synapse Analytics (vs Azure Analysis Service)

* Data warehouse and analytics platform
* Compute & Storage
  + storage and compute independently
  + Data is stored in Azure Data Lake, doesn’t have to be moved into Synapse Analytics to be analyzed
* Big data analytics platform (PaaS)
* Complex queries and ETL/ELT operations
* Data from multiple sources is combined into a single location
* Compute can be paused to reduce costs (one of the few PaaS services that can do this)
* Distributes processing across compute nodes
* Uses Online Analytical Processing (OLAP) for reporting solutions
* Can analyze data stored in Parquet format
* Linked connection to bring data in
* Ingest - > Transform - > store - > serve
* Synapse studio includes embedded Apache Spark/Synapse SQL/Synapse Pipelines
* AI and ML are also built in
* Synapse pipelines allows you to visually build your data injection and data transformation workload
* enterprise data warehouse, or EDW.
* run complex queries across petabytes of data using massively parallel processing, or MPP
* Can ingest data from Azure Data Lake Storage, files, or other DBMSs
* Using Polybase external tables and the COPY command is the fastest way to load tables into Azure Synapse Analytics via T-SQL
* Can run serverless (on-demand), only pay for queries you run on your data
* Control Node and Compute Node
  + Control node is the brain of the architecture, the front end that interacts with users and apps
  + Compute node is the computational power, the back end that does all the queries and computing

Azure Analysis Services (vs Azure Synapse Analytics)

* Processing data on a smaller scale than Azure Synapse Analytics
* build tabular models to support online analytical processing (OLAP) queries
* Data source -> Injection -> Data storage -> analysis - > Visualization
* Correlate multiple sources



Data Ingestion in Azure

* Azure Data Factory
  + Heterogenous
* SQL Server Integration Services (SSIS)
  + Heterogenous
  + Part of MS SQL Server
  + No coding necessary
  + Prepare data for SQL workloads via transformation pipelines

Data processing options

* Azure Synapse Analytics
* Azure Databricks
* Azure HDInsight
* Azure Data Factory
* Azure Data Lake Store

Jobs

* Database administrator
  + Azure Data Studio
  + SQL Server Management Studio (SSMS)
  + Azure Portal and CLI
* Data engineer
  + Azure Synapse Studio
  + SQL
  + Azure Portal and CLI
  + HDInsights
  + Azure Databricks
* Data analyst
  + Power BI

ARM templates

* a JSON file that defines the infrastructure and configuration for your project.
* allows you to create and deploy an entire Azure infrastructure declaratively
* automation

Microsoft SQL Server Management Studio (SSMS)

* Graphical tool for managing any SQL infrastructure
* Can be used to query Azure Synapse Analytics data warehouse
* Cannot enable users to create and use SQL notebooks (only Azure Data Studio can do this)
* Traditional deep management, query store, etc;
* integrated environment for managing any SQL infrastructure, from SQL Server to Azure SQL Database.

Azure Data Studio (vs SSMS)

* Cross platform db tool
* Lightweight editor
* Very similar to VS code, but web version
* Simple data queries, visualizations, etc
* Can embed documents and query results into a SQL notebook
* Can query a MS SQL Server big data cluster
* Can be used to restore a db

Microsoft Visual Studio Code

* Lightweight source code editor

Microsoft SQL Server Data Tools (SSDT) for Visual Studio

* Graphical tool that supports Offline db development
* Transforms database development
* Runs inside Visual Studio

Query SQL data from the command line

* sqlcmd
* lets you enter Transact-SQL statements, system procedures, and script files at the command prompt

SSDT vs SSMS

* Development tool (in Visual Studio) vs graphical tool

Microsoft Power BI service

* SaaS
* Report and dashboard creating
* report sharing and distribution
* Visualization of data
* Graphical representation of data
* Insights
* Direct integration w/ Azure Services
* Reports vs Dashboards
  + Dashboards are interactive
* Power BI Service vs Power BI Desktop

Visualizations in Power BI

* Bar and Column charts
* Matrix
* Line chart
* Key Influencer
* Scatter
  + display relationships between 2 (scatter) or 3 (bubble) quantitative measures -- whether or not, in which order, etc.
  + dots on a graph
* Treemap
  + charts of colored rectangles

Data Analysis

* Descriptive – what happened
* Diagnostic – why happened
* Predictive – what will happen
* Prescriptive – what should I do
* Cognitive - What if this happens
  + Conclusions based on existing knowledge