Certificate in Master Data Migration

Master Data Architecture

Attribute – a single data element that describes a property of an entity, such as Customer Name or Product Code.
Related terms: field, data element, property.
Explanation: In master data architecture, attributes are the building blocks that capture the characteristics of master entities. They are defined in the data model and stored in the underlying repository.
Example: The Customer entity may have attributes like First Name, Last Name, Email Address, and Customer Status.
Practical application: Attributes are mapped during migration to ensure that source system fields align with target system attributes.
Challenges: Inconsistent naming, differing data types, and missing attributes across source systems can cause mapping errors and data quality issues.

Canonical Data Model (CDM) – a standardized, organization-wide representation of data used to enable seamless integration between heterogeneous systems.
Related terms: enterprise data model, data abstraction, data interchange format.
Explanation: The CDM acts as a neutral reference that all source and target systems translate to and from, reducing the need for point-to-point mappings.
Example: An e-commerce firm defines a CDM for product information that includes SKU, Title, Category, and Price. All legacy product databases map their native fields to this CDM before loading into the new system.
Practical application: During master data migration, the CDM simplifies transformation logic and supports future integrations.
Challenges: Designing a CDM that accommodates all legacy variations without becoming overly complex, and keeping the CDM synchronized with evolving business requirements.

Data Architecture – the overall structural design of data assets, data flows, standards, and governance mechanisms within an organization.
Related terms: data blueprint, information architecture, technical architecture.
Explanation: It defines how data is collected, stored, processed, and accessed, providing a roadmap for migration, integration, and analytics.
Example: A retail chain's data architecture may include a data lake for raw transaction logs, an MDM hub for product master data, and a data warehouse for sales reporting.
Practical application: A clear data architecture guides the sequencing of migration activities, identifies dependencies, and ensures alignment with strategic objectives.
Challenges: Legacy landscape complexity, siloed data ownership, and rapidly changing technology stacks can impede the creation of a coherent architecture.

Data Governance – the set of policies, processes, roles, and metrics that ensure data is managed as a

valuable enterprise asset.
Related terms: data stewardship, data policy, data quality management.
Explanation: Governance establishes accountability, defines data standards, and monitors compliance throughout the data lifecycle, including migration.
Example: A governance council mandates that all master data records must have a valid Global Unique Identifier (GUID) and undergo quarterly quality checks.
Practical application: Governance frameworks provide the criteria for data validation, approval workflows, and issue escalation during migration projects.
Challenges: Gaining executive sponsorship, reconciling conflicting stakeholder priorities, and enforcing policies across autonomous business units.

Data Integration – the process of combining data from disparate sources into a unified view for analysis or operational use.
Related terms: data consolidation, data federation, ETL, ELT.
Explanation: Integration techniques vary from batch extraction-transform-load to real-time API-based synchronization, each influencing migration design.
Example: An organization extracts customer records from a legacy CRM, enriches them with loyalty data from a separate system, and loads the consolidated view into the MDM hub.
Practical application: Effective integration reduces duplication, improves data consistency, and accelerates time-to-value after migration.
Challenges: Managing schema mismatches, handling data latency, and ensuring transactional integrity across heterogeneous environments.

Data Lineage – the documented history of data's origin, movements, transformations, and destinations.
Related terms: data provenance, traceability, metadata.
Explanation: Lineage maps the flow from source fields through transformation rules to target attributes, enabling impact analysis and auditability.
Example: A lineage diagram shows that LegacySystemA.CustomerID maps to MDM.Customer.GUID after a cleansing step that removes leading zeros.
Practical application: During migration, lineage supports root-cause analysis of data anomalies and satisfies regulatory reporting requirements.
Challenges: Capturing lineage for legacy systems lacking documentation, and maintaining accurate lineage as transformation logic evolves.

Data Mapping – the specification of how source data elements correspond to target data elements, often expressed as mapping tables or scripts.
Related terms: attribute mapping, transformation mapping, data dictionary.
Explanation: Mapping defines the rules for data conversion, including field renaming, type conversion, and business logic application.
Example: Mapping rule: Source.Price (string) → Target.StandardPrice (decimal) with a conversion that strips currency symbols and applies rounding.
Practical application: Accurate mapping is critical to preserve data semantics and avoid loss of meaning during migration.

Challenges: Complex many-to-many relationships, inconsistent source definitions, and undocumented legacy transformations increase mapping effort.

Data Migration – the systematic process of moving data from one system or environment to another, while preserving quality, integrity, and relevance.
Related terms: data conversion, data transfer, cutover.
Explanation: Migration encompasses extraction, transformation, loading, validation, and post-migration reconciliation.
Example: Migrating product master data from an on-premise ERP to a cloud-based MDM platform involves extracting CSV files, applying business rules, and loading via bulk APIs.
Practical application: A well-planned migration minimizes operational disruption and ensures business continuity.
Challenges: Data volume, downtime constraints, and hidden data quality defects often cause schedule overruns and budget escalations.

Data Quality – the measure of data's accuracy, completeness, consistency, timeliness, and relevance for its intended purpose.
Related terms: data cleansing, data profiling, data validation.
Explanation: High-quality master data underpins reliable analytics, operational efficiency, and regulatory compliance.
Example: A data quality rule may require that every Customer.Email contains an "@" symbol and passes a domain validation check.
Practical application: Quality rules are embedded in migration pipelines to flag or correct records before they enter the target system.
Challenges: Legacy data may contain duplicates, missing values, and outdated formats that are costly to remediate.

Data Stewardship – the responsibility for managing data assets, ensuring they meet quality standards, and supporting business users.
Related terms: data owner, data custodian, data governance.
Explanation: Stewards act as subject-matter experts who define rules, resolve issues, and approve changes to master data.
Example: The product data steward validates new SKU entries for correct categorization and pricing hierarchy before they are loaded into the MDM hub.
Practical application: Engaging stewards early in migration projects accelerates rule definition and facilitates smoother acceptance of the new system.
Challenges: Limited time allocation for stewards, ambiguous role boundaries, and resistance to change can hinder effective stewardship.

Data Warehouse – a centralized repository optimized for query and analysis, storing integrated, subject-oriented, and historical data.
Related terms: data mart, star schema, OLAP.
Explanation: While not a master data store, the warehouse often consumes master data to enrich

transactional facts for reporting.
Example: A sales data warehouse joins transaction lines with the Customer master to produce region-level revenue dashboards.
Practical application: Accurate master data migration ensures that downstream reporting remains reliable and consistent.
Challenges: Misalignment between warehouse dimensional models and master data structures can cause join errors and inaccurate metrics.

Entity Relationship (ER) Modeling – a technique for visualizing and defining the logical relationships between data entities.
Related terms: conceptual model, relational schema, cardinality.
Explanation: ER diagrams help architects identify primary keys, foreign keys, and association rules, forming the basis for migration schema design.
Example: An ER diagram shows a one-to-many relationship between Customer (parent) and Order (child) entities.
Practical application: During migration, ER models guide the creation of referential integrity constraints in the target database.
Challenges: Translating legacy, undocumented relationships into a coherent ER model often requires extensive data profiling.

Enterprise Data Model (EDM) – a high-level, organization-wide representation of core data domains and their interconnections.
Related terms: canonical data model, logical data model, data architecture.
Explanation: The EDM provides a common vocabulary for business and IT, supporting initiatives such as master data migration, analytics, and compliance.
Example: The EDM includes domains such as Customer, Product, Supplier, and Location, each with defined attributes and relationships.
Practical application: Aligning migration scope with the EDM ensures that all critical master entities are addressed and that cross-domain dependencies are respected.
Challenges: Keeping the EDM current as business processes evolve and reconciling divergent interpretations across departments.

ETL (Extract, Transform, Load) – a batch-oriented integration pattern that extracts data from source systems, applies transformation logic, and loads the result into a target repository.
Related terms: ELT, data pipeline, batch processing.
Explanation: ETL is commonly used for large-scale master data migrations where performance and control over transformation steps are paramount.
Example: An ETL job extracts Vendor records from a legacy ERP, standardizes address formats, and loads them into the MDM hub.
Practical application: Scheduling ETL jobs during low-usage windows reduces impact on operational systems.
Challenges: Complex transformations can lead to long processing times, and error handling must be robust to avoid data loss.

ELT (Extract, Load, Transform) – an integration approach that loads raw data into a staging area first, then performs transformations within the target environment.
Related terms: ETL, in-database processing, data lake.
Explanation: ELT leverages the processing power of modern databases or data lakes, simplifying architecture and often reducing data movement.
Example: Raw customer files are loaded into a cloud data lake; SQL scripts then cleanse and enrich the data before it is moved to the MDM system.
Practical application: ELT is advantageous when the target platform offers scalable compute resources and when transformations are iterative.
Challenges: Requires careful security controls on the staging area and may expose raw data to unauthorized access if not properly governed.

Entity Governance – the set of policies and controls that dictate how specific master entities are created, updated, and retired.
Related terms: data governance, lifecycle management, data stewardship.
Explanation: Governance rules for each entity ensure consistency, prevent duplication, and enforce business constraints.
Example: The Product entity governance mandates that any new product must undergo a review process and receive an approved Category assignment before activation.
Practical application: Embedding entity governance into migration workflows automates compliance checks and reduces manual rework.
Challenges: Overly rigid rules can stall migration, while lax controls may allow poor-quality data to proliferate.

Global Unique Identifier (GUID) – a system-generated, universally unique key used to identify master records across all applications.
Related terms: surrogate key, primary key, UUID.
Explanation: GUIDs decouple business keys from technical identifiers, supporting data consolidation and interoperability.
Example: A customer record receives a GUID like 3F2504E0-4F89-11D3-9A0C-0305E82C3301, which is referenced by downstream systems instead of the legacy customer number.
Practical application: GUIDs simplify merging records from multiple source systems during migration, avoiding key collisions.
Challenges: GUIDs increase storage requirements, can be less human-readable, and may require additional indexing strategies for performance.

Metadata – data that describes other data, such as definitions, structures, lineage, and usage contexts.
Related terms: data dictionary, data catalog, data provenance.
Explanation: Metadata enables discovery, impact analysis, and governance by providing contextual information about master data assets.
Example: The metadata entry for Product.Price includes its data type (decimal), source system (ERP), and business rule (must be non-negative).
Practical application: Maintaining accurate metadata during migration helps downstream applications

interpret the data correctly.
Challenges: Legacy systems often have incomplete or inaccurate metadata, requiring extensive discovery efforts.

Reference Data – relatively static data that categorizes or classifies other data, such as country codes, currency codes, or industry classifications.
Related terms: master data, lookup tables, code lists.
Explanation: Reference data provides the contextual framework for master data and is frequently shared across multiple domains.
Example: The ISO 3166 country code list is used to standardize Customer.Country values across all systems.
Practical application: Consolidating reference data during migration reduces redundancy and ensures uniform reporting.
Challenges: Inconsistent code usage, outdated values, and lack of a single source of truth can lead to mismatches.

Source System – the original application or database that holds the master data to be migrated.
Related terms: legacy system, upstream system, origin.
Explanation: Understanding source system schemas, data quality, and business rules is essential for accurate migration planning.
Example: An on-premise SAP ERP serves as the source system for product master data.
Practical application: Direct extraction from the source system enables incremental migration and reduces the need for intermediate staging.
Challenges: Limited access permissions, proprietary data formats, and undocumented customizations can impede extraction.

Target System – the destination application or platform where master data will reside after migration.
Related terms: destination system, new system, sink.
Explanation: The target system's data model, validation rules, and performance constraints shape the migration design.
Example: A cloud-based MDM solution with a RESTful API is the target for migrated customer records.
Practical application: Aligning target system capabilities with business requirements ensures that the migrated data can be leveraged immediately.
Challenges: Incompatible data types, restrictive APIs, and capacity limits may require data transformation or staging.

Transformation Rules – the business logic applied to source data to convert it into the target format, including calculations, look-ups, and conditional mappings.
Related terms: data mapping, data cleansing, business rules.
Explanation: Rules are codified in scripts, configuration files, or mapping tools and must be thoroughly tested before execution.
Example: A rule that calculates StandardPrice as ListPrice × (1 – DiscountRate) and rounds to two decimal places.
Practical application: Centralizing transformation rules in a repository promotes reuse across multiple

migration runs.
Challenges: Complex conditional logic, dependency ordering, and performance overhead can make rule implementation difficult.

Versioning – the practice of tracking changes to master data definitions, schemas, and migration scripts over time.
Related terms: change management, baseline, revision control.
Explanation: Version control ensures that any alteration to data structures or transformation logic is auditable and reversible.
Example: A Git repository holds the Customer mapping file, with each commit documenting a new attribute addition.
Practical application: During migration, versioned scripts enable rollback to a known good state if validation failures occur.
Challenges: Inadequate documentation of versions can lead to confusion, especially when multiple teams work concurrently.

Workflow – an orchestrated sequence of tasks, approvals, and notifications that guide data through migration stages.
Related terms: process automation, approval chain, orchestration.
Explanation: Workflows enforce governance, track progress, and ensure that required quality checks are performed before data is loaded.
Example: A workflow that routes newly transformed supplier records to the data steward for validation before final load.
Practical application: Automated workflows reduce manual effort and provide audit trails for compliance.
Challenges: Designing flexible yet robust workflows that accommodate exceptions without causing bottlenecks.

Data Profiling – the systematic analysis of source data to assess its structure, content, and quality characteristics.
Related terms: data discovery, data quality assessment, statistical analysis.
Explanation: Profiling generates metrics such as null rates, distinct counts, and pattern frequencies, informing mapping and cleansing strategies.
Example: Profiling reveals that 12 % of Customer.PhoneNumber entries contain non-numeric characters, indicating a need for cleansing.
Practical application: Results guide the prioritization of data quality remediation tasks before migration.
Challenges: Large data volumes and encrypted fields can limit the depth of profiling, and profiling tools may misinterpret custom formats.

Data Cleansing – the process of detecting and correcting inaccurate, incomplete, or inconsistent data values.
Related terms: data scrubbing, data standardization, data quality improvement.
Explanation: Cleansing may involve format conversion, duplicate removal, address validation, and enrichment with external sources.

Example: Standardizing all Address.State values to two-letter abbreviations (e.g., “California” → “CA”).
Practical application: Clean data reduces downstream errors and improves the reliability of analytics after migration.
Challenges: Over-cleaning can inadvertently alter legitimate data, while under-cleaning leaves quality issues unresolved.

Data Consolidation – the act of merging data from multiple source systems into a single, unified repository.
Related terms: data integration, data aggregation, master data creation.
Explanation: Consolidation eliminates redundancy, resolves conflicts, and creates a single source of truth for each master entity.
Example: Combining customer records from three regional CRMs into one global MDM hub, applying deduplication rules.
Practical application: Consolidated master data supports consistent reporting and streamlined operations.
Challenges: Conflict resolution (e.g., differing attribute values for the same entity) requires well-defined business rules and stakeholder agreement.

Data Synchronization – the ongoing process of keeping master data consistent across multiple systems after migration.
Related terms: data replication, data federation, change data capture.
Explanation: Synchronization can be uni-directional (source to target) or bi-directional, depending on integration needs.
Example: An API-based sync updates the Product.Price in the e-commerce platform whenever the MDM hub price changes.
Practical application: Maintaining synchronization prevents data drift and ensures that downstream processes operate on current master data.
Challenges: Latency, conflict resolution, and handling offline systems complicate synchronization design.

Data Replication – the duplication of data from a primary system to one or more secondary systems, often for performance or availability reasons.
Related terms: data mirroring, standby, backup.
Explanation: Replication can be synchronous (real-time) or asynchronous (batch), influencing migration cutover strategies.
Example: Replicating the master customer table to a reporting database for fast analytics queries.
Practical application: Replication enables zero-downtime migration by allowing the new system to be populated while the old system remains active.
Challenges: Maintaining data consistency, handling schema changes, and managing replication latency.

Data Lifecycle – the sequence of stages that data passes through, from creation to archival or deletion.
Related terms: data governance, data retention, data retirement.
Explanation: Master data lifecycle management defines policies for onboarding, modification, deprecation, and disposal of master records.
Example: A product is introduced (creation), later updated with new specifications (maintenance), and eventually discontinued (retirement).

Practical application: Embedding lifecycle controls in migration workflows ensures that only active records are moved, reducing unnecessary load.
Challenges: Determining appropriate retention periods and automating lifecycle transitions without disrupting business processes.

Data Security – the set of controls and technologies that protect data from unauthorized access, alteration, or destruction.
Related terms: encryption, access control, data masking.
Explanation: Security measures must be applied both in transit during migration and at rest in the target environment.
Example: Encrypting CSV files with AES-256 before transferring them to the cloud MDM platform.
Practical application: Implementing role-based access controls on the migration tool limits exposure of sensitive fields such as SocialSecurityNumber.
Challenges: Balancing security with performance, managing encryption keys, and complying with jurisdictional data protection laws.

Data Privacy – the principle and regulatory requirement that personal data be collected, used, and disclosed in a lawful and transparent manner.
Related terms: GDPR, CCPA, data anonymization.
Explanation: Privacy considerations impact which attributes can be migrated, how they are stored, and who can access them.
Example: Masking Customer.SSN with a hash function before loading into a test environment.
Practical application: Privacy impact assessments guide the selection of data elements for migration and dictate necessary consent documentation.
Challenges: Identifying all personally identifiable information (PII) across legacy systems and ensuring compliance across multiple jurisdictions.

Data Compliance – adherence to internal policies, industry standards, and external regulations governing data handling.
Related terms: audit, regulatory reporting, data governance.
Explanation: Compliance checks are embedded in migration pipelines to verify that data transformations meet statutory requirements.
Example: Verifying that all financial transaction records include a valid TaxIdentificationNumber before they are loaded into the new ERP.
Practical application: Automated compliance validation reduces the risk of costly penalties post-migration.
Challenges: Keeping up-to-date with evolving regulations and interpreting ambiguous requirements into concrete technical controls.

Data Standardization – the act of applying uniform formats, codes, and naming conventions to data elements across the enterprise.
Related terms: data normalization, data harmonization, data quality.
Explanation: Standardization simplifies integration, improves searchability, and supports consistent reporting.

Example: Converting all dates to the ISO 8601 format (YYYY-MM-DD) regardless of source system representation.
Practical application: Standardized data reduces the need for ad-hoc conversion logic in downstream applications.
Challenges: Legacy systems may store dates in locale-specific formats, and enforcing standards may require extensive data transformation.

Data Enrichment – the process of augmenting master data with additional information from external sources to increase its value.
Related terms: data augmentation, third-party data, data enhancement.
Explanation: Enrichment can add demographic details, geocodes, or industry classifications that support analytics and decision-making.
Example: Adding latitude and longitude coordinates to Store.Location records using a geocoding service.
Practical application: Enriched master data enables advanced capabilities such as location-based marketing after migration.
Challenges: Ensuring data licensing compliance, handling mismatched keys, and maintaining enrichment updates over time.

Data Orchestration – the coordination of multiple data processing activities, often across different tools and environments, to achieve a cohesive workflow.
Related terms: workflow, pipeline, automation.
Explanation: Orchestration platforms schedule, monitor, and manage dependencies among extraction, transformation, validation, and load tasks.
Example: Using an orchestration engine to trigger the ETL job, followed by a data quality validation step, and finally a notification to the data steward.
Practical application: Centralized orchestration provides visibility into migration progress and facilitates rapid issue resolution.
Challenges: Integrating heterogeneous tools, handling error propagation, and ensuring scalability for large data volumes.

Data Integration Platform – a software suite that provides capabilities for connecting, transforming, and delivering data across systems.
Related terms: middleware, iPaaS, enterprise service bus.
Explanation: Platforms may offer pre-built connectors, visual mapping, and governance features that accelerate migration projects.
Example: An integration platform as a service (iPaaS) offering connectors for SAP, Salesforce, and the target MDM API.
Practical application: Leveraging a platform reduces custom coding effort and provides built-in monitoring and logging.
Challenges: Licensing costs, vendor lock-in, and the need to customize connectors for unique legacy interfaces.

Batch Processing – the execution of data operations on large groups of records at scheduled intervals rather

than in real time.
Related terms: ETL, offline processing, bulk load.
Explanation: Batch jobs are suitable for migrating massive data sets when downtime windows are available.
Example: A nightly batch extracts all new and changed product records, applies transformations, and loads them into the MDM hub.
Practical application: Batch processing allows for thorough validation before committing data to production.
Challenges: Long processing times can extend migration windows, and error handling must be robust to prevent partial loads.

Real-time Processing – the handling of data as it is generated or received, with minimal latency.
Related terms: streaming, event-driven architecture, CDC.
Explanation: Real-time pipelines are used when master data must be instantly available to downstream systems, such as in e-commerce.
Example: A change data capture (CDC) mechanism streams new customer registrations into the MDM hub within seconds.
Practical application: Real-time processing enables continuous synchronization, reducing the need for large batch cutovers.
Challenges: Requires high-availability infrastructure, sophisticated error handling, and careful management of data consistency.

API (Application Programming Interface) – a set of defined methods and data structures that allow applications to interact programmatically.
Related terms: web service, REST, SOAP.
Explanation: APIs are commonly used to load or retrieve master data during migration, especially for cloud-based targets.
Example: Posting a JSON payload to the MDM /customers endpoint to create a new master record.
Practical application: APIs provide granular control, enabling incremental migration and real-time validation of each record.
Challenges: Rate limits, authentication complexities, and differing data format expectations can slow migration progress.

Data Governance Council – a cross-functional body that defines, approves, and oversees data governance policies and initiatives.
Related terms: steering committee, data policy, governance framework.
Explanation: The council sets the strategic direction for master data management, including migration standards and timelines.
Example: The council approves the master data migration charter, defines acceptable data quality thresholds, and assigns stewardship responsibilities.
Practical application: Council decisions are documented and referenced throughout the migration to ensure alignment with organizational objectives.
Challenges: Coordinating schedules of senior leaders, reconciling divergent department priorities, and maintaining momentum over long-term projects.

Data Policy – a formal statement that outlines the rules, responsibilities, and expectations for data handling within the organization.
Related terms: governance, standards, compliance.
Explanation: Policies cover aspects such as data ownership, retention, access, and quality requirements for master data.
Example: A policy that mandates all master data changes be captured in an audit log with user attribution.
Practical application: Policies provide the criteria against which migration activities are measured and audited.
Challenges: Translating high-level policy language into actionable technical controls and ensuring consistent enforcement.

Data Strategy – the long-term plan that aligns data initiatives with business goals, outlining how data will be used as a strategic asset.
Related terms: roadmap, vision, enterprise data management.
Explanation: The strategy defines priorities for master data creation, migration, analytics, and governance.
Example: The data strategy prioritizes the migration of customer master data to support a new omnichannel experience.
Practical application: A clear strategy guides resource allocation, technology selection, and timeline planning for the migration project.
Challenges: Shifting business priorities, budget constraints, and rapid technology change can require frequent strategy revisions.

Data Architecture Blueprint – a detailed diagrammatic representation of the target data environment, including layers, components, and interfaces.
Related terms: data architecture, solution design, reference architecture.
Explanation: The blueprint serves as a contract between business and technical teams, illustrating how master data will be stored, accessed, and governed.
Example: The blueprint shows the MDM hub, downstream data lake, and analytics layer, with data flow arrows indicating integration points.
Practical application: Using the blueprint during migration helps validate that each component is provisioned correctly and that data flows as intended.
Challenges: Maintaining the blueprint's accuracy as design decisions evolve and ensuring that all stakeholders interpret it consistently.

Data Lake – a centralized repository that stores raw, unstructured, and structured data at scale, often on inexpensive storage.
Related terms: data hub, data reservoir, big data.
Explanation: While not a master data store, a lake can hold source extracts for staging, archival, or exploratory analysis before loading into MDM.
Example: Raw CSV dumps of legacy customer files are landed in an S3-based data lake for preprocessing.
Practical application: Leveraging a data lake enables flexible, schema-on-read processing and supports iterative migration testing.
Challenges: Without proper governance, lakes can become "data swamps" with unmanaged, low-quality

data that hinders downstream consumption.

Data Hub – a central integration point that consolidates, cleanses, and distributes master data to consuming applications.
Related terms: MDM hub, data integration, data sharing.
Explanation: The hub enforces governance rules, provides a single source of truth, and often includes workflow and stewardship capabilities.
Example: A hub receives product data from ERP, supplier portals, and third-party catalogs, harmonizes attributes, and publishes the consolidated view via APIs.
Practical application: Migrating master data into a hub creates a reusable asset that feeds multiple downstream systems, reducing duplication.
Challenges: Designing the hub's data model to accommodate diverse source structures and managing change as business needs evolve.

Data Vault – a modeling technique that captures all data changes over time, emphasizing auditability and scalability.
Related terms: hub-link-satellite, historical data, data warehousing.
Explanation: In a vault, entities are stored as hubs, relationships as links, and descriptive attributes as satellites, preserving raw source information.
Example: A Customer hub stores the business key, while a satellite records each change to address or status with timestamps.
Practical application: Using a vault for migration provides a complete history, supporting regulatory audits and rollback capabilities.
Challenges: The vault's highly normalized structure can increase query complexity and may require additional transformation for reporting consumption.

Data Mart – a focused subset of a data warehouse tailored to a specific business line or function.
Related terms: data warehouse, dimensional model, subject area.
Explanation: Data marts often contain denormalized, ready-to-use data for reporting and may rely on master data for dimension consistency.
Example: A sales data mart includes fact tables for orders and a dimension table for customers sourced from the MDM hub.
Practical application: Ensuring that migrated master data aligns with data mart dimensions prevents mismatched reporting.
Challenges: Maintaining synchronization between the central master data store and multiple data marts, especially after incremental loads.

Data Federation – a virtual integration approach that provides a unified view of data without moving it from its source locations.
Related terms: data virtualization, query layer, composite view.
Explanation: Federation enables users to query master data across systems as if it were a single repository, useful during phased migration.
Example: A federation layer combines customer records from ERP, CRM, and a cloud SaaS platform into a

single virtual table.
Practical application: Federation can serve as a temporary bridge while legacy systems are decommissioned, reducing the need for immediate full migration.
Challenges: Performance overhead, inconsistent security models across sources, and limited support for complex transformations.

Data Virtualization – the creation of abstracted data services that expose data from multiple sources through a common interface, often in real time.
Related terms: data federation, abstraction layer, data services.
Explanation: Virtualization decouples consumption from physical storage, allowing applications to access master data without replication.
Example: A virtual data service presents a unified Customer view that pulls attributes from both an on-premise database and a cloud CRM.
Practical application: Virtualization supports gradual migration strategies, enabling new applications to consume data from the target system while legacy sources remain active.
Challenges: Managing latency, ensuring data consistency, and handling schema evolution across the virtualized sources.

Data Modeling Techniques – the set of methodologies used to design logical and physical data structures, such as ER modeling, dimensional modeling, and object-oriented modeling.
Related terms: schema design, normalization, denormalization.
Explanation: Selecting appropriate techniques influences migration complexity, performance, and future extensibility.
Example: Using a star schema for analytical reporting while maintaining a normalized relational model for the master data repository.
Practical application: Aligning modeling techniques with business use cases ensures that migrated data supports both operational and analytical needs.
Challenges: Balancing the trade-offs between normalization (reducing redundancy) and denormalization (improving query speed) in the target architecture.

Dimensional Modeling – a design approach that structures data into fact tables and dimension tables to support analytical queries.
Related terms: star schema, snowflake schema, data warehouse.
Explanation: While primarily used for analytical data, dimensional models often reference master data dimensions for consistency.
Example: A SalesFact table includes a foreign key to the CustomerDim dimension populated from the MDM hub.
Practical application: Ensuring that dimensional keys are correctly populated during migration prevents orphaned records and inaccurate reporting.
Challenges: Maintaining synchronization between dimension records and the source master data, especially when dimensions are refreshed incrementally.

Normalized Model – a relational design that organizes data to minimize redundancy and dependency,

following normal forms (1NF, 2NF, 3NF, etc.).
Related terms: ER modeling, data integrity, relational schema.
Explanation: Normalization promotes data consistency and supports efficient updates, making it a common choice for master data repositories.
Example: Storing address components in a separate Address table linked to Customer via a foreign key.
Practical application: Normalized structures simplify enforcement of referential integrity during migration.
Challenges: Complex joins may impact query performance, and some reporting tools prefer denormalized views,