

2026-10-03

Table des matières

1 Data Requirements Specification - Template	2	1.7 Data Integration Requirements	8
1.1 Data Terminology	2	1.7.a Integration Types and Considerations	8
1.2 Data Modeling Requirements	3	1.7.b Gherkin Format for Integration Requirements	8
1.2.a Entity-Relationship Modeling Format	3	1.7.c Example Integration Requirement	9
1.2.b Entity Template	3	1.8 Data Migration and Transformation Requirements	9
1.2.c Example Entity Documentation	3	1.8.a Migration and Transformation Considerations	9
1.3 Data Quality Requirements	3	1.8.b Gherkin Format for Migration Requirements	10
1.3.a Data Quality Dimensions	4	1.8.c Example Migration Requirement	10
1.3.b Gherkin Format for Data Quality Requirements	4	1.9 Data Dependency Framework	10
1.3.c Example Data Quality Requirement	4	1.9.a Data Dependency Types	10
1.4 Data Governance Requirements	5	1.9.b Dependency Documentation Format	11
1.4.a Key Data Governance Areas ...	5	1.9.c Example Data Dependency Documentation	11
1.4.b Gherkin Format for Data Governance Requirements	5	1.10 Data Requirement Traceability ...	11
1.4.c Example Data Governance Requirement	5	1.10.a Data Requirement Traceability Matrix	11
1.5 Data Compliance Requirements	6	1.10.b Data Requirement Tags	12
1.5.a Common Regulatory Frameworks	6	1.10.c Example with Data Traceability	12
1.5.b Gherkin Format for Compliance Requirements	6	1.11 Data Requirement Authoring Guidelines	12
1.5.c Example Data Compliance Requirement	6	1.12 Gherkin Tag Reference for Data Requirements	12
1.6 Data Storage and Persistence Requirements	7	1.12.a Data Quality Tags	13
1.6.a Key Storage Requirement Categories	7	1.12.b Data Modeling Tags	13
1.6.b Gherkin Format for Storage Requirements	7	1.12.c Data Storage Tags	13
1.6.c Example Storage Requirement .	7	1.12.d Data Security Tags	14
		1.12.e Data Integration Tags	14

1.12.f	Data Migration Tags	14
1.12.g	Compliance and Regulatory Tags	15
1.12.h	Priority and Status Tags	15
1.12.i	Testing Tags	15
1.12.j	Usage Examples	16
1.13	Entity-Relationship Diagrams (ERDs)	16
1.13.a	ERD Notation Standards	16
1.13.b	Text-Based ERDs with PlantUML	17
1.13.c	ERD Best Practices	18
1.13.d	Layered ERD Documentation	18

1 Data Requirements Specification - Template

1.1 Data Terminology

Term	Definition
Entity	A distinct object, concept, or thing about which data is stored (e.g., Customer, Order, Product).
Attribute	A characteristic or property of an entity (e.g., customer_name, order_date).
Relationship	An association between two or more entities (one-to-one, one-to-many, many-to-many).
Primary Key	A unique identifier for a record in a database table.
Foreign Key	A field that links to the primary key of another table, establishing a relationship.

Term	Definition
Cardinality	Defines the numerical relationship between entities (e.g., 1:1, 1:N, N:M).
Data Type	The classification of data that determines the possible values and operations on the data.
Schema	A logical container for database objects that defines the structure of the data.
Normalization	The process of organizing data to reduce redundancy and improve data integrity.
Denormalization	The process of combining normalized tables to optimize read performance.
Data Dictionary	A centralized repository of information about data, including meaning, relationships, origin, and format.
CRUD	Create, Read, Update, Delete - the four basic operations on persistent data.
Data Validation	The process of ensuring data meets specified criteria before being processed or stored.
Data Migration	The process of transferring data between storage systems, formats, or applications.
Data Persistence	The ability of data to outlive the process that created it, typically through storage in databases or files.

1.2 Data Modeling Requirements

Data modeling is the process of creating a visual representation of data structures and their relationships within a system. This section outlines the approach to documenting data modeling requirements.

1.2.a Entity-Relationship Modeling Format

When documenting entity-relationship models, include the following information:

1. **Entity Name:** The name of the entity, typically a noun
2. **Entity Description:** A brief description of what the entity represents
3. **Attributes:** List of attributes with their data types, constraints, and descriptions
4. **Primary Key:** Identification of primary key attribute(s)
5. **Relationships:** Connections to other entities with cardinality notation
6. **Constraints:** Business rules or validation requirements for the entity

1.2.b Entity Template

```
Entity: [Entity_Name]
Description: [Brief description of the entity]
Attributes:
  - [attribute_name]: [data_type] [constraints] - [description]
  - [attribute_name]: [data_type] [constraints] - [description]
Primary Key: [attribute_name(s)]
Relationships:
  - [relationship_type] to [related_entity] ([cardinality])
Constraints:
  - [constraint_description]
```

1.2.c Example Entity Documentation

```
Entity: User
Description: Represents a registered user of the application
Attributes:
  - user_id: UUID NOT NULL - Unique identifier for the user
  - username: VARCHAR(50) NOT NULL UNIQUE - User's chosen username
  - email: VARCHAR(100) NOT NULL UNIQUE - User's email address
  - password_hash: VARCHAR(255) NOT NULL - Hashed representation of the user's password
  - created_at: TIMESTAMP NOT NULL - Date and time when the user account was created
  - last_login: TIMESTAMP - Date and time of the user's most recent login
Primary Key: user_id
Relationships:
  - One-to-Many to UserProfile (1:1)
  - One-to-Many to Post (1:N)
  - Many-to-Many to Group (N:M)
Constraints:
  - Email must be in valid format
  - Password must meet complexity requirements (8+ chars, mixture of letters, numbers, and symbols)
```

1.3 Data Quality Requirements

Data quality requirements specify the characteristics that data must possess to be considered accurate, reliable, and suitable for its intended purpose. These requirements are critical for ensuring that data-driven decisions are based on sound information.

1.3.a Data Quality Dimensions

Dimension	Description	Example Requirement
Accuracy	The extent to which data correctly reflects the real-world entity it represents	User age must be calculated precisely from date of birth
Completeness	The degree to which all required data is present	All mandatory product attributes must be present before publication
Consistency	The absence of contradictions within the data	User address details must be consistent across all related tables
Timeliness	How up-to-date the data is relative to its use case	Transaction data must be no more than 5 minutes old when displayed
Validity	Adherence to defined formats, types, and value ranges	Phone numbers must follow the E.164 international format
Uniqueness	The absence of duplicate records	Each customer must have exactly one unique record

Dimension	Description	Example Requirement
Integrity	Maintenance of data relationships and constraints	Foreign key constraints must be enforced for all entity relationships

1.3.b Gherkin Format for Data Quality Requirements

```
Feature: E[XX]-F[XX] - [Data Quality Feature]
```

```
    @data-quality    @[dimension]
    @priority-level
```

```
    Scenario: E[XX]-F[XX]-S[XX] - [Specific data quality requirement]
```

```
        Given [initial data state]
```

```
        When [data operation or evaluation is performed]
```

```
        Then [expected data quality outcome]
```

```
        And [additional verification]
```

1.3.c Example Data Quality Requirement

```
Feature: E02-F01 - Product Data Quality
```

```
    @data-quality    @completeness
    @critical
```

```
    Scenario: E02-F01-S01 - Mandatory Product Attributes
```

```
        Given a new product is being created
```

```
        When the product data is submitted
```

```
        Then all mandatory attributes should be present
```

```
        And the system should prevent saving if any mandatory attribute is missing
```

```

@data-quality @validity @high
Scenario: E02-F01-S02 - Product
Price Validation
    Given product data includes price
    information
    When the product data is validated
    Then the price must be greater
    than zero
    And the price must be rounded to
    two decimal places

```

1.4 Data Governance Requirements

Data governance requirements define how an organization administers, manages, and protects its data assets. These requirements cover policies, processes, roles, and standards for data management.

1.4.a Key Data Governance Areas

Area	Description
Data Ownership	Who is responsible for specific data domains
Access Control	Who can access what data and under what circumstances
Data Life-cycle	How data progresses from creation to archival or deletion
Data Classification	How data is categorized based on sensitivity and value
Metadata Management	How data about data is captured and maintained
Data Standards	Format, syntax, and structure conventions
Data Stewardship	Roles and responsibilities for data quality and management

1.4.b Gherkin Format for Data Governance Requirements

```

Feature: E[XX]-F[XX] - [Data
Governance Feature]

    @data-governance @[area]
    @[priority]
    Scenario: E[XX]-F[XX]-S[XX] -
    [Specific governance requirement]
    Given [context for governance
    rule]
    When [action related to data]
    Then [expected governance
    enforcement]

```

1.4.c Example Data Governance Requirement

```

Feature: E03-F01 - Personal Data
Governance

    @data-governance @access-control
    @critical
    Scenario: E03-F01-S01 - Personal
    Data Access Restrictions
    Given personal user data is stored
    in the system
    When a staff member attempts to
    access user personal data
    Then access should be granted only
    if the staff member has appropriate
    permissions
    And all access attempts should be
    logged for audit purposes

    @data-governance @data-lifecycle
    @high
    Scenario: E03-F01-S02 - Personal
    Data Retention Policy
    Given a user has been inactive
    for 24 months
    When the data cleanup process runs
    Then the user's personal data

```

should be anonymized
And a record of the anonymization
should be maintained

1.5 Data Compliance Requirements

Data compliance requirements ensure that data handling practices adhere to legal regulations, industry standards, and organizational policies. These requirements are particularly important for personal, financial, or otherwise sensitive data.

1.5.a Common Regulatory Frameworks

Framework	Description	Key Requirements
GDPR	EU General Data Protection Regulation	Lawful basis for processing, consent management, data subject rights
CCPA/CPRA	California Consumer Privacy Act/Rights Act	Consumer rights, opt-out mechanisms, data sale restrictions
HIPAA	Health Insurance Portability and Accountability Act	PHI protection, security measures, breach notification
PCI DSS	Payment Card Industry Data Security Standard	Secure processing, transmission, and storage of payment data
SOX	Sarbanes-Oxley Act	Financial data integrity, audit

Framework	Description	Key Requirements
		trails, security controls

1.5.b Gherkin Format for Compliance Requirements

Feature: E[XX]-F[XX] - [Compliance Feature]

@compliance @[regulation]
@[priority]

Scenario: E[XX]-F[XX]-S[XX] - [Specific compliance requirement]

Given [compliance context]

When [relevant action occurs]

Then [expected compliant behavior]

1.5.c Example Data Compliance Requirement

Feature: E04-F01 - GDPR Compliance

@compliance @gdpr @critical

Scenario: E04-F01-S01 - User Data Export Request

Given a verified user has submitted a data export request

When the request is processed

Then all personal data associated with the user should be compiled

And the data should be provided in a machine-readable format

And the export should be completed within 30 days of the request

@compliance @gdpr @critical

Scenario: E04-F01-S02 - Right to be Forgotten

Given a verified user has submitted a deletion request

When the request is processed

Then all personal data should be permanently deleted

And any data required for legal purposes should be isolated and minimized

And confirmation of deletion should be provided to the user

1.6 Data Storage and Persistence Requirements

This section defines the requirements for how data should be stored, persisted, and physically managed within the system, including database technologies, caching strategies, and storage formats.

1.6.a Key Storage Requirement Categories

Category	Description	Examples
Database Selection	Requirements for database technology	SQL vs. NoSQL, specific vendors, cloud vs. on-premises
Storage Models	Data organization and representation	Relational, document, key-value, graph, time-series
Persistence Strategies	How and when data is saved	Immediate vs. batch, transaction boundaries
Caching	Requirements for data caching	TTL values, invalidation strategies, cache layers

Category	Description	Examples
File Storage	Requirements for file/blob storage	Object storage, file systems, compression
Archival	Long-term storage of historical data	Retention policies, storage tiers, retrieval SLAs
Backup	Data backup requirements	Frequency, retention, recovery time objectives (RTO)

1.6.b Gherkin Format for Storage Requirements

```
Feature: E[XX]-F[XX] - [Storage Feature]
```

```
@storage @[category] @[priority]
Scenario: E[XX]-F[XX]-S[XX] - [Specific storage requirement]
Given [storage context]
    When [relevant action or condition]
    Then [expected storage behavior]
```

1.6.c Example Storage Requirement

```
Feature: E05-F01 - User Data Storage
```

```
@storage @persistence @high
Scenario: E05-F01-S01 - User Profile Image Storage
    Given a user uploads a profile image
    When the image is processed
```

Then the original image should be stored in cloud object storage

And a resized thumbnail should be cached for quick access

And the image URL should be stored in the user database record

```
@storage @backup @critical
Scenario: E05-F01-S02 - User Data Backup
    Given user data exists in the primary database
    When the scheduled backup process runs
    Then a full backup should be created daily
    And incremental backups should be created hourly
    And backups should be retained for 90 days
```

1.7 Data Integration Requirements

Data integration requirements specify how data flows between different systems, applications, or components, including APIs, ETL processes, and real-time data synchronization.

1.7.a Integration Types and Considerations

Type	Description	Key Considerations
API Integration	Real-time data exchange via APIs	API formats, authentication, rate limiting
ETL Processes	Extract, Transform, Load for batch processing	Schedule, data mapping, error handling

Type	Description	Key Considerations
Event-based Integration	Data change triggered by events	Event formats, ordering, delivery guarantees
File-based Integration	Data change via files	File formats, transfer protocols, validation
Database-level Integration	Direct database access or replication	Schema compatibility, transaction boundaries
Message Queues	Asynchronous data exchange via messaging	Message formats, queuing systems, processing guarantees
Middleware	Software that bridges between systems	Connection protocols, transformation rules

1.7.b Gherkin Format for Integration Requirements

```
Feature: E[XX]-F[XX] - [Integration Feature]
```

```
@integration @[type] @[priority]
Scenario: E[XX]-F[XX]-S[XX] -
```

```
[Specific integration requirement]
  Given [integration context]
  When [integration event or action]
  Then [expected integration outcome]
```

1.7.c Example Integration Requirement

Feature: E06-F01 - Payment Gateway Integration

```
@integration @api @critical
  Scenario: E06-F01-S01 - Payment Processing
    Given a user submits a payment for an order
    When the payment request is sent to the payment gateway
    Then the request should use OAuth 2.0 for authentication
    And the transaction should be processed within 3 seconds
    And the order status should be updated based on the payment response
```

```
@integration @event-based @high
  Scenario: E06-F01-S02 - Inventory Update After Purchase
    Given a successful payment is processed
    When the order confirmation event is published
    Then the inventory system should receive the event within 1 second
    And the product stock levels should be updated accordingly
    And any low stock conditions should trigger replenishment events
```

1.8 Data Migration and Transformation Requirements

This section outlines requirements for moving data from legacy systems to new systems, or

transforming data between different formats or structures.

1.8.a Migration and Transformation Considerations

Aspect	Description	Key Requirements
Source Data Analysis	Understanding the current data	Completeness, quality assessment, volume
Data Mapping	Defining how source data maps to target	Field mappings, transformations, business rules
Data Cleansing	Improving data quality during migration	De-duplication, normalization, enrichment
Migration Strategy	Approach to moving the data	Big bang vs. phased, cutover planning
Validation & Verification	Ensuring migration accuracy	Reconciliation, sampling, quality checks
Roll-back Plan	Recovery approach if migration fails	Backup strategy, reversion process
Timeline & Performance	Speed and timing considerations	Downtime window, processing throughput

1.8.b Gherkin Format for Migration Requirements

```
Feature: E[XX]-F[XX] - [Migration Feature]

  @migration @[aspect] @[priority]
  Scenario: E[XX]-F[XX]-S[XX] - [Specific migration requirement]
    Given [migration context]
    When [migration process or step]
    Then [expected migration outcome]
```

1.8.c Example Migration Requirement

```
Feature: E07-F01 - Customer Data Migration

  @migration @data-mapping @high
  Scenario: E07-F01-S01 - Customer Address Standardization
    Given customer data is being migrated from the legacy system
    When address information is processed
    Then addresses should be standardized to a consistent format
    And geographic coordinates should be added for each address
    And incomplete addresses should be flagged for review

  @migration @validation @critical
  Scenario: E07-F01-S02 - Migration Data Reconciliation
    Given the customer data migration has completed
    When the validation process runs
    Then the total number of customers should match between old and new systems
    And a sample of 10% of records should be verified for accuracy
```

And any discrepancies should be logged and reviewed

1.9 Data Dependency Framework

Understanding and documenting data dependencies is crucial for managing changes, troubleshooting issues, and ensuring data integrity across systems.

1.9.a Data Dependency Types

Dependency Type	Description	Example
Entity Dependency	One entity requiring another	Order depends on Customer and Product
Attribute Dependency	Values derived from other attributes	Total Price depends on Item Price and Quantity
System Dependency	Data sourced from external systems	Shipping rates from carrier API
Calculation Dependency	Values derived through calculation	Tax amount based on price and location

Dependency Type		Description	Example
Temporal dependency	De-	Time-based dependencies	Monthly report depends on daily aggregations
Conditional dependency	De-	Dependencies based on conditions	Discount applies only if quantity > 10

1.9.b Dependency Documentation Format

```

Data Element: [Element_Name]
Description: [Description of data element]
Type: [Entity/Attribute/Calculation/etc.]
Dependencies:
  - Depends on: [Dependency_1] - [Relationship description]
  - Depends on: [Dependency_2] - [Relationship description]
Dependents:
  - Used by: [Dependent_1] - [Relationship description]
  - Used by: [Dependent_2] - [Relationship description]

```

1.9.c Example Data Dependency Documentation

```

Data Element: OrderTotal
Description: The total monetary value of an order
Type: Calculation
Dependencies:

```

```

- Depends on: LineItemSubtotals - Sum of all line item subtotals
- Depends on: ShippingCost - Cost to ship the order
- Depends on: TaxAmount - Tax applied to the order
Dependents:
- Used by: InvoiceAmount - Forms the basis for invoice creation
- Used by: RevenueReports - Included in revenue calculations
- Used by: CustomerSpending - Contributes to customer lifetime value

```

1.10 Data Requirement Traceability

Traceability for data requirements helps ensure that all data needs are addressed throughout the development lifecycle and can be traced back to their origins.

1.10.a Data Requirement Traceability Matrix

Data Req ID	Source Business Requirement	Data Model	Implementation	Data Quality Test	Status
DR-001	Client Meeting #3	Cus- tome- rEn- ti- ty	Custom- erEn- ti- ty	Data Valid- ation- Test	Ver- i- fied
DR-002	Re- BR-047	Au- dit- Lo- gEn- ti- ty	Audit Log- gEn- ti- ty	Com- plian- ceTest	In- Pro- gress

1.10.b Data Requirement Tags

Use these special tags to establish traceability in data requirements:

Tag	Purpose	Example
<code>@source-document</code>	Links to requirement source document	<code>@source-document:data-spec-v1.2</code>
<code>@business-req</code>	Links to business requirement	<code>@business-req:BR-023</code>
<code>@data-model</code>	Links to data model	<code>@data-model:CustomerEntity</code>
<code>@schema</code>	Links to database schema	<code>@schema:public.customers</code>
<code>@implementation</code>	Links to code implementation	<code>@implementation:com.example.data.CustomerRepository</code>

1.10.c Example with Data Traceability

```
@data-quality @completeness
@critical
@source-document:customer-data-spec
@business-req:BR-023
@data-model:CustomerEntity
@schema:public.customers
@implementation:com.example.data.CustomerValidator

Scenario: DR-001 - Customer Contact Information Completeness

  Given a new customer record is being created
  When the record is validated before saving
  Then at least one contact method (email, phone, or address) must be present
  And all provided contact information must pass format validation
```

1.11 Data Requirement Authoring Guidelines

To ensure clarity, consistency, and completeness in data requirements documentation, follow these guidelines:

1. **Be Specific About Data Characteristics:** Clearly specify data types, formats, ranges, and constraints
2. **Use Examples:** Provide concrete examples of valid and invalid data
3. **Define Relationships Explicitly:** Clearly articulate relationships between data entities
4. **Specify Cardinality:** Include the numerical relationships between entities (1:1, 1:N, N:M)
5. **Include Validation Rules:** Document all rules for validating data integrity
6. **Document Derivations:** Explain how calculated or derived fields should be computed
7. **Address Edge Cases:** Consider and document how to handle nulls, duplicates, and exceptions
8. **Use Standard Notation:** Use ERD notations, UML, or other standard formats when applicable
9. **Link to Glossary:** Reference the project glossary for all domain-specific terms
10. **Include Data Volume Estimates:** Document expected data volumes and growth projections

1.12 Gherkin Tag Reference for Data Requirements

This reference table provides standardized tags for data-related requirements, organized by category. Using these tags consistently will improve requirement organization, filtering, and traceability.

1.12.a Data Quality Tags

Tag	Description	Example Usage
<code>@data-quality</code>	General tag for data quality requirements	<code>@data-quality</code> <code>@completeness</code>
<code>@accuracy</code>	Requirements for data correctness	<code>@data-quality</code> <code>@accuracy</code>
<code>@completeness</code>	Requirements for required data presence	<code>@data-quality</code> <code>@completeness</code>
<code>@consistency</code>	Requirements for data coherence across entities	<code>@data-quality</code> <code>@consistency</code>
<code>@timeliness</code>	Requirements for data freshness	<code>@data-quality</code> <code>@timeliness</code>
<code>@validity</code>	Requirements for data format and value constraints	<code>@data-quality</code> <code>@validity</code>
<code>@uniqueness</code>	Requirements for data deduplication	<code>@data-quality</code> <code>@uniqueness</code>
<code>@integrity</code>	Requirements for maintaining relationships	<code>@data-quality</code> <code>@integrity</code>

1.12.b Data Modeling Tags

Tag	Description	Example Usage
<code>@entity</code>	Requirements for entity definitions	<code>@entity</code> <code>@user</code>
<code>@relationship</code>	Requirements for entity relationships	<code>@relationship</code> <code>@one-to-many</code>

Tag	Description	Example Usage
<code>@attribute</code>	Requirements for specific attributes	<code>@attribute</code> <code>@sensitive</code>
<code>@primary-key</code>	Requirements for primary key definitions	<code>@primary-key</code> <code>@natural-key</code>
<code>@foreign-key</code>	Requirements for foreign key associations	<code>@foreign-key</code> <code>@cascade</code>
<code>@constraint</code>	Requirements for data constraints	<code>@constraint</code> <code>@check</code>
<code>@normalization</code>	Requirements for data normalization	<code>@normalization</code> <code>@third-normal-form</code>

1.12.c Data Storage Tags

Tag	Description	Example Usage
<code>@storage</code>	General tag for storage requirements	<code>@storage</code> <code>@persistence</code>
<code>@persistence</code>	Requirements for data persistence strategies	<code>@persistence</code> <code>@immediate</code>
<code>@database</code>	Database-specific requirements	<code>@database</code> <code>@relational</code>
<code>@sql</code>	SQL database requirements	<code>@sql</code> <code>@postgres</code>
<code>@nosql</code>	NoSQL database requirements	<code>@nosql</code> <code>@document-db</code>
<code>@caching</code>	Caching strategy requirements	<code>@caching</code> <code>@invalidation</code>

Tag	Description	Example Usage
@file-storage	File and blob storage requirements	@file-storage @object-store
@backup	Backup strategy requirements	@backup @incremental
@archival	Archival strategy requirements	@archival @cold-storage

1.12.d Data Security Tags

Tag	Description	Example Usage
@data-security	General tag for data security	@data-security @encryption
@encryption	Data encryption requirements	@encryption @at-rest
@masking	Data masking requirements	@masking @partial
@tokenization	Data tokenization requirements	@tokenization @pii
@access-control	Access control requirements	@access-control @rbac
@audit	Audit trail requirements	@audit @access-logs
@anonymization	Data anonymization requirements	@anonymization @irreversible
@pseudonymization	Data pseudonymization requirements	@pseudonymization @key-mapping

1.12.e Data Integration Tags

Tag	Description	Example Usage
@integration	General tag for data integration	@integration @api
@api	API integration requirements	@api @rest
@etl	ETL process requirements	@etl @batch
@event-based	Event-driven integration requirements	@event-based @pub-sub
@file-based	File transfer integration requirements	@file-based @sftp
@messaging	Messaging system requirements	@messaging @queue
@sync	Synchronous integration requirements	@sync @real-time
@async	Asynchronous integration requirements	@async @eventual-consistency

1.12.f Data Migration Tags

Tag	Description	Example Usage
@migration	General tag for data migration	@migration @data-mapping
@data-mapping	Data mapping requirements	@data-mapping @field-transformation
@data-cleansing	Data cleansing requirements	@data-cleansing @deduplication
@cutover	Migration cutover requirements	@cutover @big-bang

Tag	Description	Example Usage
<code>@validation</code>	Migration validation requirements	<code>@validation</code> <code>@reconciliation</code>
<code>@rollback</code>	Migration rollback requirements	<code>@rollback</code> <code>@point-in-time</code>
<code>@data-conversion</code>	Data format conversion requirements	<code>@data-conversion</code> <code>@encoding</code>
<code>@data-enrichment</code>	Data enrichment requirements	<code>@data-enrichment</code> <code>@third-party</code>

1.12.g Compliance and Regulatory Tags

Tag	Description	Example Usage
<code>@compliance</code>	General tag for compliance requirements	<code>@compliance</code> <code>@gdpr</code>
<code>@gdpr</code>	GDPR compliance requirements	<code>@gdpr</code> <code>@right-to-access</code>
<code>@ccpa</code>	CCPA/CPRA compliance requirements	<code>@ccpa</code> <code>@opt-out</code>
<code>@hipaa</code>	HIPAA compliance requirements	<code>@hipaa</code> <code>@phi</code>
<code>@pci</code>	PCI DSS compliance requirements	<code>@pci</code> <code>@card-data</code>
<code>@sox</code>	Sarbanes-Oxley compliance requirements	<code>@sox</code> <code>@audit-trail</code>
<code>@retention</code>	Data retention requirements	<code>@retention</code> <code>@time-limited</code>

Tag	Description	Example Usage
<code>@delete</code>	Data deletion requirements	<code>@deletion</code> <code>@hard-delete</code>

1.12.h Priority and Status Tags

Tag	Description	Example Usage
<code>@critical</code>	Critical priority requirements	<code>@critical</code> <code>@data-security</code>
<code>@high</code>	High priority requirements	<code>@high</code> <code>@data-quality</code>
<code>@medium</code>	Medium priority requirements	<code>@medium</code> <code>@reporting</code>
<code>@low</code>	Low priority requirements	<code>@low</code> <code>@enhancement</code>
<code>@implemented</code>	Implemented requirements	<code>@implemented</code> <code>@version-1.2</code>
<code>@verified</code>	Verified requirements	<code>@verified</code> <code>@test-suite-3</code>
<code>@wip</code>	Work in progress requirements	<code>@wip</code> <code>@sprint-4</code>
<code>@blocked</code>	Blocked requirements	<code>@blocked</code> <code>@dependency</code>

1.12.i Testing Tags

Tag	Description	Example Usage
<code>@automated-test</code>	Requirements with automated tests	<code>@automated-test</code> <code>@unit</code>
<code>@manual-test</code>	Requirements needing manual testing	<code>@manual-test</code> <code>@exploratory</code>
<code>@validation-test</code>	Data validation test requirements	<code>@validation-test</code> <code>@boundary</code>

Tag	Description	Example Usage
@performance-test	Performance test requirements	@performance-test @volume
@scalability-test	Scalability test requirements	@scale-test @load
@recovery-test	Recovery test requirements	@recovery-test @failover
@corruption-test	Data corruption test requirements	@corruption-test @repair

1.12.j Usage Examples

Here are examples of how to combine these tags effectively:

```
@data-quality @completeness
@critical @automated-test
Scenario: DR-001 - Validating
Customer Record Completeness
    Given a new customer record is being
    submitted
    When validation is performed
    Then all required fields must be
    present
```

```
@data-security @encryption @pci @high
@compliance
Scenario: DR-002 - Credit Card Data
Storage
    Given a payment includes credit
    card information
    When the data is persisted
    Then the card number must be
    tokenized
    And the CVV must never be stored
```

```
@migration @data-mapping @data-
cleansing @medium @wip
Scenario: DR-003 - Customer Address
```

Migration

```
Given legacy addresses are being
migrated
    When the migration process runs
    Then addresses should be parsed into
    standardized components
    And invalid postal codes should be
    flagged for review
```

For effective tagging, combine tags from different categories to provide a comprehensive classification of each requirement.

1.13 Entity-Relationship Diagrams (ERDs)

Entity-Relationship Diagrams are essential tools for visualizing data structures, relationships, and cardinality in data requirements specifications. They provide a clear representation of how different data entities relate to each other and what attributes they possess.

1.13.a ERD Notation Standards

Multiple notation systems exist for ERDs. The most common are:

1.13.a.a Chen Notation

- **Entities:** Represented by rectangles
- **Attributes:** Represented by ovals connected to entities
- **Relationships:** Represented by diamonds connecting entities
- **Cardinality:** Written alongside relationship lines

1.13.a.b Crow's Foot Notation

- **Entities:** Represented by rectangles
- **Attributes:** Listed inside the entity rectangle
- **Relationships:** Represented by lines between entities
- **Cardinality:** Indicated by symbols at the endpoints of relationship lines:

- One (1): A vertical line |
- Many (N): A crow's foot >|<
- Zero or One (0..1): O|
- Zero or Many (0..N): O|<
- One or Many (1..N): |<

1.13.a.c UML Class Diagram Notation

- **Entities:** Represented by rectangles divided into sections
- **Attributes:** Listed in the middle section of the entity
- **Operations:** Listed in the bottom section of the entity
- **Relationships:** Represented by lines with different arrowheads
- **Cardinality:** Written as ranges (1, 0..1, , 1..) near the ends of relationship lines

1.13.b Text-Based ERDs with PlantUML

PlantUML provides a simple syntax for creating ERDs using the Crow's Foot notation:

```
@startuml
' Define entities with their attributes
entity "Customer" as customer {
    * customer_id : UUID <<PK>>
    --
    * name : VARCHAR(100)
    * email : VARCHAR(100) <<unique>>
    created_at : TIMESTAMP
}

entity "Order" as order {
    * order_id : UUID <<PK>>
    --
    * customer_id : UUID <<FK>>
    * order_date : TIMESTAMP
    * status : VARCHAR(20)
    total_amount : DECIMAL(10,2)
}

entity "Product" as product {
    * product_id : UUID <<PK>>
```

```
--
* name : VARCHAR(100)
* price : DECIMAL(10,2)
description : TEXT
stock_quantity : INTEGER
}

entity "OrderItem" as orderItem {
    * order_id : UUID <<PK,FK>>
    * product_id : UUID <<PK,FK>>
    --
    * quantity : INTEGER
    unit_price : DECIMAL(10,2)
    subtotal : DECIMAL(10,2)
}

' Define relationships with cardinality
customer ||--o{ order : places
order ||--o{ orderItem : contains
product ||--o{ orderItem : included in

@enduml
```

1.13.b.a PlantUML ERD Relationship Notation

Relationship Type	Notation	Description
One-to-Many	entity1 --o{ entity2	Entity1 has one or more entity2
Many-to-Many	entity1 } o--o{ entity2	Entity1 has many entity2, and vice versa
One-to-One	entity1 -- entity2	Entity1 has exactly one entity2

Relationship Type		Notation	Description
Zero-or-One Many	to	<code>entity1</code> <code> 0--o{ entity2</code>	Entity1 has zero or one relationship to many entity2
Zero-or-One One	to	<code>entity1</code> <code> 0-- entity2</code>	Entity1 has zero or one relationship to exactly one entity2

1.13.c ERD Best Practices

1. **Focus on Entities First:** Start by identifying the main entities before adding attributes or relationships
2. **Use Meaningful Names:** Entity and attribute names should clearly reflect what they represent
3. **Mark Key Fields:** Clearly identify primary keys (PK) and foreign keys (FK)
4. **Include Cardinality:** Always specify the numerical relationship between entities
5. **Group Related Entities:** Arrange the diagram so that related entities are positioned near each other
6. **Add Constraints:** Document any important constraints on attributes or relationships
7. **Keep It Simple:** Avoid cluttering the diagram - consider creating multiple focused ERDs for complex data models
8. **Consistent Notation:** Use the same notation style throughout your documentation
9. **Include Data Types:** Specify the data types for attributes, especially in technical ERDs
10. **Document Mandatory Attributes:** Use notation (such as * in PlantUML) to indicate required attributes

1.13.d Layered ERD Documentation

For complex systems, consider a layered approach to ERD documentation:

1. **Conceptual ERD:** High-level view showing only major entities and relationships, without attributes
2. **Logical ERD:** More detailed view with all entities, relationships, and attributes but platform-independent
3. **Physical ERD:** Implementation-specific diagram with database-specific data types, constraints, and indices