

The **User** relation represents system users who create accounts to save starred articles, receive alerts, and track updates. Email functionally depends on UserID and must be unique.

User			
PK	UserID	INT	
Unique	Email	VARCHAR	
	Name	VARCHAR	
	CreatedAt	DATETIME	

Diagram Key

Users

Database

Products

This acts as a **many-to-many** relation between User and NewsArticle. A user may star many articles, and an article may be starred by many users.

StarredArticles			
FK	UserID	INT	
FK	ArticleId	INT	
	TimeStarred	DATETIME	

The **Source** relation contains metadata about news providers: newspapers, agencies, blogs, etc.

Source			
PK	SourceId	INT	
	Category	VARCHAR(63)	
	Name	VARCHAR(63)	
	Country	VARCHAR(63)	

BCNF

We first attain 2NF form, by putting Manufacturer and its pertaining data in its own relation. We then adhere to 3NF by having every FD be of the form SuperKey -> Prime Attribute. The last rule we follow puts us in BCNF form and its that their exist a candidate key on the left side of our FD's

The **NewsArticle** relation stores scraped articles from RSS feeds, websites, and APIs. Each article belongs to exactly **one** source and may have **one** analysis.

NewsArticle			
PK	ArticleId	INT	
	Title	TEXT	
	URL	TEXT	
FK	SourceId	INT	
	PublishedAt	DATETIME	
	Content	TEXT	

The **Analysis** relation stores AI-generated insights: sentiment, toxicity, category, keywords, bias flags, etc. Each NewsArticle may have at **most one** analysis.

Analysis			
PK	AnalysisId	INT	
FK	ArticleId	INT	
	Sentiment	ENUM	
	Toxicity	DECIMAL	
	Category	VARCHAR	
	Keywords	TEXT	
	RiskFlags	VARCHAR	

The **RiskScore** relation represents the scoring output from your weighting system: political sensitivity, corruption risk, violence flags, and probability scores.

RiskScore			
PK	ScoreId	INT	
FK	ArticleId	INT	
	OverallScore	DECIMAL	
	Factors	TEXT	
	GeneratedAt	DATETIME	

This table stores AI-detected similar or contextually linked articles

RelatedArticles			
PK	RelationId	INT	
FK	ArticleId	INT	
FK	RelatedId	INT	
	SimilarityScore	DECIMAL	

https://lucid.app/lucidchart/7ac96534-75d0-402d-97f9-794afc761ba9/edit?view_items=2A0OH.6YPnRN&page=0_0&invitationId=inv_8d1de7d6-f088-49ba-9531-5c609c8034b7

Assumptions

From left to right, the first relation in our schema is the User relation, which contains accounts for researchers who save articles, receive alerts, and track important events. Each user is uniquely identified by a UserID, and each user must have a unique Email. This relation connects to the StarredArticles junction table, which enables a many-to-many relationship between users and articles.

This feature allows researchers to “follow” an event, track updates over time, and receive notifications as new articles appear. This design supports the project goal: to help researchers write reports faster and allow the company to react more quickly to real-world events.

The StarredArticles relation acts as a junction table facilitating this many-to-many relationship. A single user may star many articles, and any article may be starred by many users. The TimeStarred attribute allows the system to determine recent priorities, trending topics, and which users should receive notifications when related events occur.

The Source relation contains metadata about news providers, such as newspapers, agencies, and online platforms. Each article must belong to exactly one source, but one source may contribute many articles. This ensures consistency in tracking which outlets frequently report on certain events or political topics.

The NewsArticle relation stores articles scraped from RSS feeds, websites, and APIs. Each article contains text content, publication date, URL, and category/detected keywords. Each article must belong to exactly one source and may optionally have one AI analysis and one risk score.

The Analysis relation stores AI-generated interpretations of each article, including sentiment, toxicity, category, keywords, and any detected red-flag patterns. Each NewsArticle may have at most one associated Analysis. This ensures a simple 1–1 mapping and avoids conflicting AI insights.

The RiskScore relation represents the weighting system that assigns political sensitivity, corruption risk, violence flags, and probability scores to each article. Like Analysis, each article may have at most one RiskScore. This score provides an at-a-glance numerical representation of the risk level behind events.

Finally, the RelatedArticles relation stores pairs of articles that are determined to be similar by the AI model. This relation is many-to-many, since any article can be related to multiple other articles. The SimilarityScore helps the alerting system recommend “other relevant articles” at the bottom of a report, one of the new features requested.

Normalization

User(UserID, Email, Name, CreatedAt)

Functional Dependencies:

$\text{UserID} \rightarrow \text{Email, Name, CreatedAt}$

$\text{Email} \rightarrow \text{UserID}$ (email must be unique)

1NF: No repeating attributes; atomic values.

2NF: No partial dependencies (PK is a single attribute).

3NF: UserID is a superkey; all non-keys depend on it.

BCNF: Left side of FD contains a candidate key \rightarrow valid.

StarredArticles(UserID, ArticleID, TimeStarred)

(Composite key: {UserID, ArticleID})

FDs:

{UserID, ArticleID} → TimeStarred

1NF: All attributes atomic.

2NF: No partial dependency; composite key is minimal.

3NF: Composite key is a superkey, all attributes depend on it.

BCNF: The key itself determines TimeStarred → valid.

Source(SourceID, Category, Name, Country)

FDs:

SourceID → Category, Name, Country

1NF: Atomic values.

2NF: PK is single attribute.

3NF: SourceID is a superkey.

BCNF: SourceID is a candidate key → valid.

NewsArticle(ArticleID, Title, URL, SourcedID, PublishedAt, Content)

FDs:

ArticleID → Title, URL, SourcedID, PublishedAt, Content

1NF: Atomic values under each column.

2NF: Single-attribute PK.

3NF: ArticleID is a superkey.

BCNF: ArticleID is a candidate key.

No transitive dependencies exist.

Analysis(AnalysisID, ArticleID, Sentiment, Toxicity, Category, Keywords, RiskFlags)

FDs:

AnalysisID → ArticleID, Sentiment, Toxicity, Category, Keywords, RiskFlags

Also, because each article may have at most one analysis:

ArticleID → AnalysisID

1NF: Atomic attributes.

2NF: Single-attribute PK.

3NF: No transitive dependencies.

BCNF: Both determinants (AnalysisID and ArticleID) are keys → valid.

RiskScore(ScoreID, ArticleID, OverallScore, Factors, GeneratedAt)

FDs:

ScoreID \rightarrow ArticleID, OverallScore, Factors, GeneratedAt

Also, at-most-one rule gives:

ArticleID \rightarrow ScoreID

1NF / 2NF / 3NF: No partial or transitive dependencies.

BCNF: Both ScoreID and ArticleID are keys \rightarrow valid.

RelatedArticles(RelationID, ArticleID, RelatedToID, SimilarityScore)

FDs:

RelationID \rightarrow ArticleID, RelatedToID, SimilarityScore

1NF: All atomic.

2NF: PK is single attribute.

3NF: No non-key determines another non-key.

BCNF: RelationID is a candidate key.

Database Schema

User(

 UserID INT PK,
 Email VARCHAR UNIQUE,
 Name VARCHAR,
 CreatedAt DATETIME

)

Source(

 SourceID INT PK,
 Category VARCHAR(63),
 Name VARCHAR(63),
 Country VARCHAR(63)

)

NewsArticle(

 ArticleID INT PK,
 Title TEXT,
 URL TEXT,
 SourcedID INT FK \rightarrow Source(SourceID),
 PublishedAt DATETIME,
 Content TEXT

)

```
StarredArticles(  
  UserID    INT          FK → User(UserID),  
  ArticleID INT          FK → NewsArticle(ArticleID),  
  TimeStarred DATETIME,  
  PRIMARY KEY(UserID, ArticleID)  
)
```

```
Analysis(  
  AnalysisID INT          PK,  
  ArticleID  INT UNIQUE   FK → NewsArticle(ArticleID),  
  Sentiment  ENUM,  
  Toxicity   DECIMAL,  
  Category   VARCHAR,  
  Keywords   TEXT,  
  RiskFlags  VARCHAR  
)
```

```
RiskScore(  
  ScoreID    INT          PK,  
  ArticleID  INT UNIQUE   FK → NewsArticle(ArticleID),  
  OverallScore DECIMAL,  
  Factors     TEXT,  
  GeneratedAt DATETIME  
)
```

```
RelatedArticles(  
  RelationID INT          PK,  
  ArticleID  INT          FK → NewsArticle(ArticleID),  
  RelatedToID INT         FK → NewsArticle(ArticleID),  
  SimilarityScore DECIMAL  
)
```