Design and Normalization Database on SuperMarket Aisle Management

Fadhla Mohamed Mutua - SM3201434

19 giugno 2025

Sommario

Questo rapporto presenta il progetto concettuale e logico di un database per la gestione delle corsie di un supermercato. Il sistema gestisce supermercati, corsie, articoli, produttori, distanze e la registrazione degli errori relativi agli articoli. Vengono spiegate le operazioni SQL utilizzate (vedi codice allegato), i diagrammi Entità-Relazione (ER), la ridondanza e le fasi di normalizzazione fino alla Terza Forma Normale (3NF).

Indice

1 Introduzione

Questo rapporto riguarda un sistema di database che supporta supermercati e produttori registrando gli errori generati da trigger automatici.

Assunzione: più supermercati possono avere una corsia con lo stesso nome, pertanto usando `AisleID` si garantisce che indipendentemente dal nome della corsia, l'`AisleID` corrisponda a uno specifico supermercato.

Nota: La relazione tra `Aisle` e `SuperMarket` nel PDF è chiamata *Has Aisle* mentre nel codice allegato è denominata `AisleSuperMarket`.

2. Panoramica delle operazioni SQL

Questa sezione delinea tutte le operazioni SQL principali implementate nel sistema di database, spiegandone il ruolo e come supportano la logica del dominio per la gestione e la conformità delle corsie.

2.1 Creazione delle Tabelle

Operazione: Definizione e inizializzazione dello schema

Il primo insieme di operazioni definisce la struttura relazionale e lo schema del database. Tabelle come `SuperMarket`, `Aisle`, `Item`, `Producer`, `Distance`, `Contain`, `Manufactured_By`, `ErrorMessages` e `ItemLogErrors` sono create usando `CREATE TABLE`. Queste istruzioni specificano chiavi primarie e esterne, tipi di dati e vincoli, stabilendo la base per l'archiviazione dei dati.

2.2 Trigger

Trigger: `trg_check_Item_Aisle_count`

Questo trigger `BEFORE INSERT` garantisce che un articolo non sia posizionato in più corsie all'interno dello stesso supermercato. In caso di violazione genera un'eccezione SQL. Ciò assicura un'associazione uno a uno tra articolo e corsia per supermercato.

Trigger: `trg_log_item_wrong_aisle`

Questo trigger `AFTER INSERT` verifica se l'articolo è stato collocato nella corsia corretta utilizzando una funzione. In caso di non conformità, registra la violazione tramite un ID errore, usando messaggi standardizzati e timestamp.

Trigger: `ReturnItem`

Monitora gli articoli inseriti per scadenza. Se un articolo è scaduto, controlla se la distanza dal produttore è entro una soglia o se l'articolo è non deperibile. Registra quindi se l'articolo deve essere restituito o scartato.

2.3 Viste

Vista: `FullItemDetails` — Risultato completo con join che mostra ogni articolo, la sua corsia, il supermercato e il produttore.

Vista: `ItemWithProducers` — Visualizza articoli e informazioni associate ai produttori.

Vista: `ProducerSuperMarketDistance` — Mappa le distanze tra produttori e supermercati.

Vista: `WhereToStore` — Abbina i tipi di conservazione degli articoli con i nomi delle corsie per la validazione della collocazione.

Vista: `ItemErrorDetails` — Fornisce registri dettagliati degli errori sugli articoli, inclusi messaggi e flag di scarto.

2.4 Funzioni memorizzate

Funzione: `fn_validate_aisle_compliance`

Applica le regole di dominio per la collocazione nelle corsie. Restituisce un errore leggibile dall'uomo se non conforme, altrimenti `NULL`.

Funzione: `fn_insert_into_error_message`

Inserisce un nuovo messaggio nella tabella `ErrorMessages` se non esiste e restituisce l'`ErrorID`.

Funzione: `fn_suggest_correct_aisle`

Suggerisce l'`AisleID` più appropriato per un articolo in uno specifico supermercato basandosi su categoria e conservazione.

2.5 Procedure memorizzate

Procedura: `pr_insert_item_log` — Inserisce registrazioni di errori sugli articoli con timestamp e informazioni sull'errore.

Procedura: `AddItemToAisle` — Aggiunge un articolo a una corsia, verificando che la corsia appartenga al supermercato specificato.

Procedura: `RemoveItemFromSuperMarket` — Elimina un articolo da tutte le corsie di un supermercato.

Procedura: `LogItemError` — Registra manualmente un errore sull'articolo usando messaggi e dettagli dell'articolo.

Procedura: `CleanExpiredItems` — Elimina articoli scaduti e deperibili dalla tabella `Contain`.

Procedura: `CheckItemCompliance` — Verifica la conformità e opzionalmente suggerisce la collocazione corretta.

Procedura: `sp_check_item_placement` — Scorre gli articoli, valida la collocazione e registra le non conformità.

Procedura: `sp_expiration_check` — Registra problemi relativi alla scadenza includendo informazioni su produttore e stato di scarto.

2.6 Eventi schedulati

Evento: `ev_daily_item_placement_check` — Trigger giornaliero per eseguire `sp_check_item_placement`.

Evento: `ev_daily_expiration_and_cleanup` — Pulizia giornaliera degli articoli scaduti e registrazione dello stato di scadenza.

Evento: `ev_daily_expiration_process` — Evento wrapper che gestisce l'automazione di scadenza e pulizia.

2.7 Query

Query: Analisi storica degli errori

```
SELECT
    em.ErrorMessage,
    i.ItemName, i.ItemStorageType,
    a.AisleID, a.AisleName AS IncorrectAisle,
    le.LogTime,
    le.ToBeThrown
FROM ItemLogErrors le
JOIN Item i ON le.ItemID = i.ItemID
JOIN Aisle a ON le.AisleID = a.AisleID
LEFT JOIN ErrorMessages em ON le.ErrorID = em.ErrorID
ORDER BY le.LogTime DESC;
```

Fornisce un registro completo delle violazioni di collocazione o scadenza degli articoli con informazioni contestuali.

2 Schema concettuale

2.1 Tabella dello schema concettuale

Entità e Relazioni	**Cardinalità (Relazionale)**
Producer — Distance — SuperMarket	1:N : 1:N
SuperMarket — Has_Aisle — Aisle	1:N : 1:1
Aisle — Contains — Item	1:1 : 0:N
Producer — Manufactured_By — Item	1:N : 1:1
ItemLogErrors — Logs_Item — Item	0:N : 0:1
ItemLogErrors — Logs_Aisle — Aisle	0:N : 0:1
ItemLogErrors — Logs_ErrorMessage — ErrorMessage	1:N : 1:1

Tabella 1: Relazioni dello schema concettuale con cardinalità precise

Relazione	Descrizione
Distance	Collega i produttori ai supermercati con informazioni sulla distanza, utile per calcolare la logistica di trasporto e della catena di approvvigionamento (in questo caso per decidere se restituire o scartare un articolo).
Has_Aisle	Associa le corsie ai supermercati, indicando la struttura interna di ciascun negozio.
Contains	Rappresenta gli articoli conservati nelle corsie, collegando l'inventario alla posizione.
Manufactured_By	Collega gli articoli ai loro produttori, permettendo la tracciabilità dell'origine del prodotto.
Logs_Item	Registra gli errori relativi ad articoli specifici, facilitando il tracciamento e la verifica degli errori.
Logs_Aisle	Registra gli errori associati a corsie specifiche, aiutando a individuare problemi di conservazione.
Logs_ErrorMessage	Associa i registri di errori a messaggi di errore standardizzati, permettendo una segnalazione coerente degli errori.

Tabella 2: Relazioni dello schema concettuale e loro descrizione

2.2 Diagramma Entità-Relazione

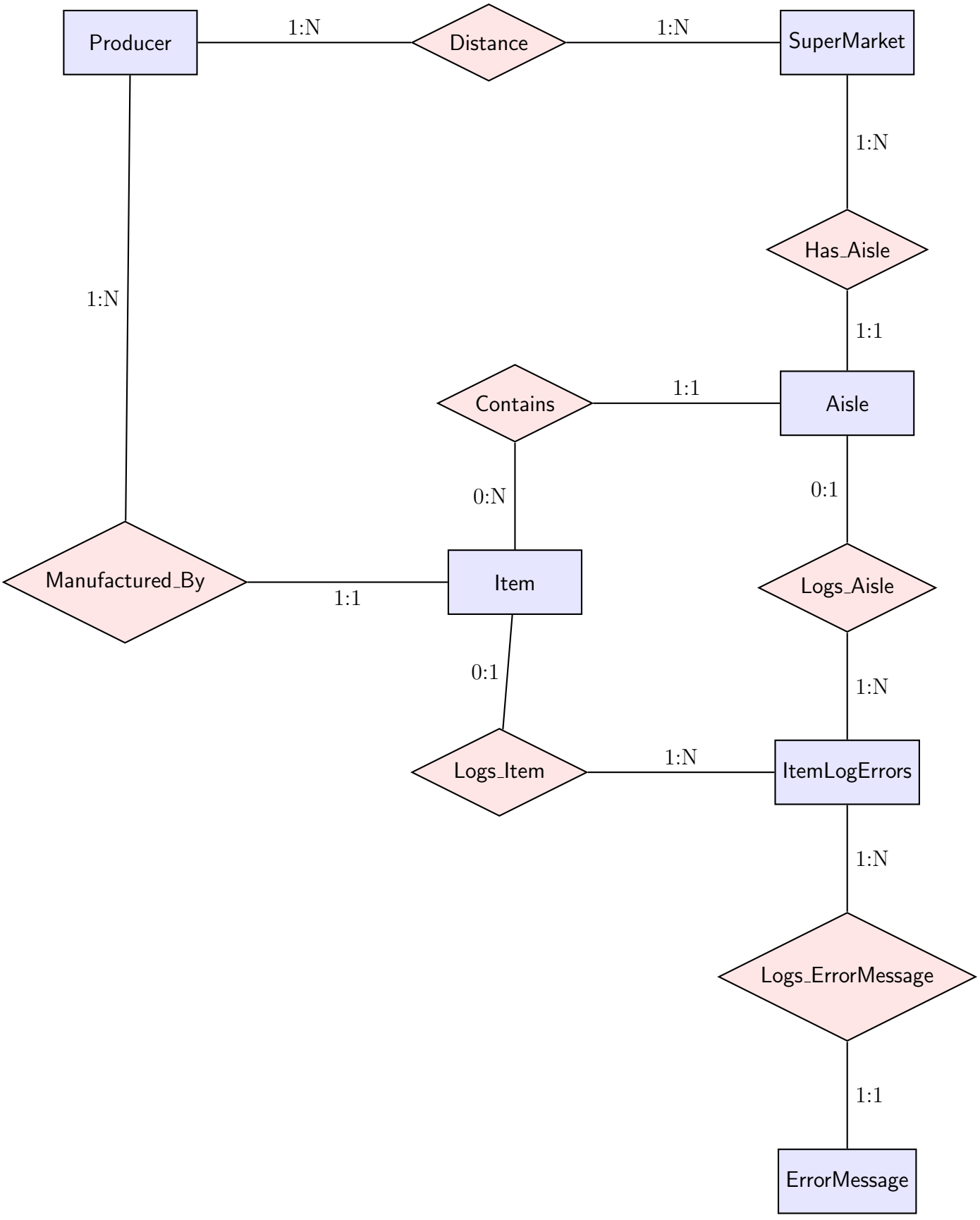

Figura 1: Diagramma Entità-Relazione per la gestione delle corsie

2.3 Analisi della ridondanza

- **ItemLogErrors** ha molteplici relazioni binarie con `Item`, `Aisle` e `ErrorMessage`.
- Potrebbe essere sostituito da una relazione ternaria che collega direttamente `ItemLogErrors` con `Item`, `Aisle` e `ErrorMessage`.
- Poiché `ItemLogErrors` è generato da trigger, mantenere relazioni separate facilita le interrogazioni.
- Non ci sono entità o relazioni ridondanti per `Producer`, `SuperMarket`, `Aisle`, `Item` o `Distance`.

3 Schema logico

Tabella	**Attributi**	**PK**	**FK**
Producer	ProducerID, ProducerName, ProducerLocation	ProducerID	-
SuperMarket	SuperMarketID, SuperMarketName, SuperMarketLocation	SuperMarketID	-
Aisle	AisleID, AisleName	AisleID	-
Item	ItemID, ItemName, ItemCategory, ItemStorageType, ItemPerishable, ItemExpirationDate	ItemID	-
Manufactured_By	ItemID, ProducerID	ItemID	ProducerID
Distance	ProducerID, SuperMarketID, Distance	ProducerID, SuperMarketID	ProducerID, SuperMarketID
Contain	AisleID, ItemID	AisleID, ItemID	AisleID, ItemID
Has_Aisle	AisleID, SuperMarketID,	AisleID, SuperMarketID,	AisleID, SuperMarketID,
ItemLogErrors	ErrorLogID, ItemID, AisleID, ErrorID, LogTime, ToBeThrown	ErrorLogID	ItemID, AisleID, ErrorID
ErrorMessage	ErrorID, ErrorMessage	ErrorID	-

Tabella 3: Tabelle dello schema logico con chiavi primarie sottolineate

4 Normalizzazione

4.1 Prima forma normale (1NF)

- Tutti gli attributi sono atomici e indivisibili.
- Non esistono array ripetuti poiché `ErrorMessage` è una stringa.

4.2 Seconda forma normale (2NF)

- Non esistono dipendenze parziali su chiavi composte.

4.3 Terza forma normale (3NF)

- Non sono presenti dipendenze transitive.
- Tutti gli attributi non chiave dipendono unicamente dalla chiave primaria.
- Esempio: in `Item`, tutti gli attributi (categoria, flag di deperibilità, scadenza) dipendono solo da `ItemID`.
- Non vengono memorizzati campi derivati o calcolati, evitando ridondanza.

Entità e Relazioni	**Cardinalità (può/(o) deve : quantità : può/(o) deve : quantità)**
Producer — Distance — SuperMarket	1:N : 1:N
SuperMarket — Has_Aisle — Aisle	1:N : 1:1
Aisle — Contains — Item	1:1 : 0:N
Producer — Manufactured_By — Item	1:N : 1:1
ItemLogErrors — Logs_Item — Item	0:N : 0:1
ItemLogErrors — Logs_Aisle — Aisle	0:N : 0:1
ItemLogErrors — Logs_ErrorMessage — ErrorMessage	1:N : 1:1

Tabella 4: Schema concettuale normalizzato con cardinalità precise

Tabella	**Attributi**	**PK**	**FK**
Producer	ProducerID, ProducerName, ProducerLocation	ProducerID	-
SuperMarket	SuperMarketID, SuperMarketName, SuperMarketLocation	SuperMarketID	-
Aisle	AisleID, AisleName	AisleID	-
Item	ItemID, ItemName, ItemCategory, ItemStorageType, ItemPerishable, ItemExpirationDate	ItemID	-
Manufactured_By	ItemID, ProducerID	ItemID	ProducerID
Distance	ProducerID, SuperMarketID, Distance	ProducerID, SuperMarketID	ProducerID, SuperMarketID
Contain	AisleID, ItemID	AisleID, ItemID	AisleID, ItemID
Has_Aisle	AisleID, SuperMarketID,	AisleID, SuperMarketID,	AisleID, SuperMarketID,
ItemLogErrors	ErrorLogID, ItemID, AisleID, ErrorID, LogTime, ToBeThrown	ErrorLogID	ItemID, AisleID, ErrorID
ErrorMessage	ErrorID, ErrorMessage	ErrorID	-

Tabella 5: Schema logico normalizzato

5 Conclusione

Il processo di normalizzazione conferma che lo schema aderisce alla 3NF, supportando il tracciamento automatico degli errori senza introdurre ridondanze.