EPICODE

Approfondimento

SQL
Modello
Entità - Relazione

Agenda

In questo approfondimento affronteremo:

- La progettazione di una base dati e le sue fasi.
- La progettazione concettuale.
- La progettazione logica.
- La progettazione fisica.
- L'importanza della chiave di lettura.
- Applicazioni dello schema ER oltre la progettazione iniziale.

Il **modello entità-relazione** è stato formalizzato nel 1976 da Peter Chen ed è un modello teorico ampiamente utilizzato nella progettazione dei DataBase, che si concretizza poi nel **diagramma entità-relazione**.

La progettazione

La progettazione di una base dati avviene solitamente in tre fasi:

1. Progettazione concettuale.
2. Progettazione logica.
3. Progettazione fisica.

Il **modello entità-relazione** va a soddisfare i requisiti della fase di progettazione concettuale e logica.

La progettazione

A livello di progettazione concettuale, il modello **entità-relazione** si caratterizza per la possibilità di astrarre tutti i dettagli superflui per definire le nostre **entità**, che rappresentano oggetti del mondo reale, e relazioni, che rappresentano le connessioni tra questi oggetti.

Nel momento dell'implementazione pratica del modello, le **entità e le relazioni** si concretizzeranno nelle tabelle e nelle **chiavi esterne** del nostro DataBase (anche le relazioni possono essere rappresentate da tabelle).

Progettazione Concettuale

Per effettuare una **progettazione concettuale** approfondita possiamo seguire questi step:

Identificare le Entità:

- **Definizione**: Identificare e definire gli oggetti o entità principali all'interno del contesto del database.
- Queste entità rappresentano oggetti, concetti o eventi del mondo reale.

  Esempio: In un database universitario, le entità potrebbero includere Studente, Corso, Professore e Dipartimento.

Identificare le Relazioni:

- **Definizione**: Determinare le relazioni tra le entità identificate. Le relazioni rappresentano come le entità sono connesse o associate tra loro.
- **Esempio**: Un'entità Studente potrebbe avere una relazione con l'entità Corso per rappresentare l'iscrizione.

Definire Attributi:

- **Definizione:** Specificare gli attributi (proprietà o caratteristiche) di ciascuna entità. Gli attributi descrivono i dettagli di un'entità e forniscono le informazioni necessarie per distinguere un'istanza da un'altra.
- **Esempio:** Gli attributi per l'entità Studente potrebbero includere Matricola, Nome e Data di Nascita.

Progettazione Concettuale

Identificare Attributi Chiave:

- **Definizione**: Identificare gli attributi chiave per ciascuna entità, che identificano in modo univoco le istanze di quell'entità. Gli attributi chiave sono cruciali per garantire l'integrità dei dati e stabilire relazioni.
- **Esempio:** La Matricola dello Studente potrebbe essere l'attributo chiave per l'entità Studente.

Affinare le Relazioni:

- **Definizione**: Affinare e specificare la natura e la cardinalità delle relazioni. La cardinalità descrive quante istanze di un'entità sono collegate a istanze di un'altra entità.
- **Esempio:** Un Corso potrebbe avere una relazione con l'entità Professore, e la cardinalità potrebbe essere "uno-a-molti" per indicare che un professore può insegnare più corsi.

Modellare i Vincoli:

- **Definizione**: Identificare e modellare eventuali vincoli o regole aziendali che si applicano alle relazioni o entità. I vincoli garantiscono che il database rimanga consistente con scenari del mondo reale.
- **Esempio:** Un vincolo potrebbe essere che uno studente non può essere iscritto a più di una sezione dello stesso corso durante un semestre specifico.

Progettazione Concettuale

Verificare e Convalidare il Modello:

- **Definizione**: Rivedere il modello concettuale per accuratezza e completezza. Assicurarsi che rifletta accuratamente i requisiti informativi dell'organizzazione o del sistema.
- Esempio: Convalidare il modello con gli interessati per confermare che sia in linea con la loro comprensione dei processi aziendali.

Documentare il Modello Concettuale:

- **Definizione**: Creare documentazione che includa diagrammi, descrizioni testuali e qualsiasi informazione aggiuntiva necessaria per comunicare il modello concettuale agli interessati.
- Esempio: Utilizzare i Diagrammi Entità-Relazione (ERD) per rappresentare visivamente entità, relazioni e attributi.

Progettazione Logica

Passando poi alla **progettazione logica**, queste sono le fasi fondamentali:

Tradurre Entità e Relazioni:

- **Definizione**: Tradurre le entità, relazioni e attributi dal modello concettuale in tabelle, colonne e relazioni di chiave esterna nel modello logico.
- **Esempio:** Ogni entità diventa una tabella, gli attributi diventano colonne e le relazioni sono rappresentate attraverso chiavi esterne.

Normalizzare le Tabelle:

- **Definizione**: Applicare tecniche di normalizzazione per minimizzare la ridondanza e i problemi di dipendenza nello schema del database. La normalizzazione aiuta a garantire l'integrità dei dati e a ridurre la possibilità di anomalie di aggiornamento.
- **Esempio:** Suddividere grandi tabelle in tabelle più piccole e correlate per eliminare dati ridondanti.

Progettazione Logica

Definire Chiavi Primarie e Chiavi Esterne:

Definizione: Identificare e definire chiavi primarie per ciascuna tabella per identificare in modo univoco i record. Stabilire relazioni di chiave esterna tra le tabelle per garantire l'integrità referenziale.

Esempio: In una tabella "Studenti", "*Matricola*" potrebbe essere la chiave primaria, e in una tabella "Corsi", "*IDCorso*" potrebbe essere la chiave primaria, con una relazione di chiave esterna alla tabella "*Studenti*".

Impostare Vincoli di Integrità:

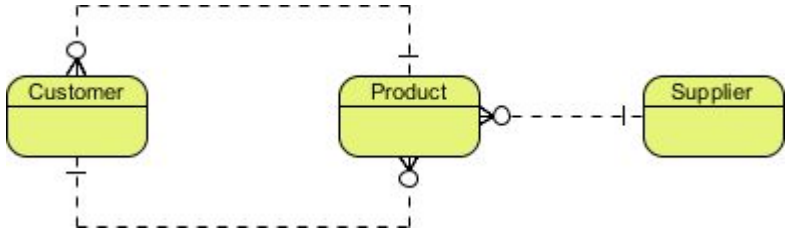
Definizione: Configurare vincoli di integrità, come vincoli univoci e vincoli di controllo, per garantire l'accuratezza e la coerenza dei dati.

Esempio: Imporre che ciascuno studente abbia una Matricola univoca e che una data di nascita rientri in un intervallo valido.

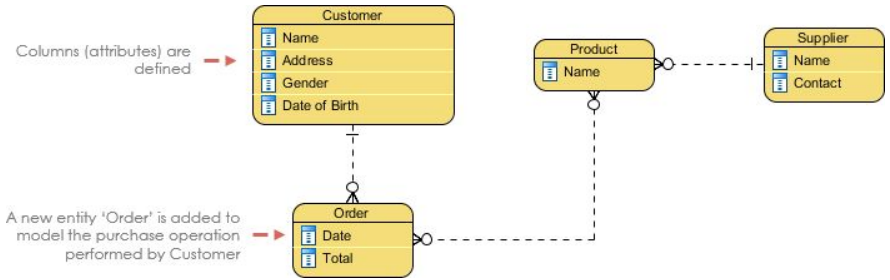
Diagramma Entità - Relazioni

La concretizzazione di queste fasi, prima dell'effettiva implementazione fisica della base dati, si concretizza nel **diagramma entità-relazioni**, in inglese ***entity-relationship diagram***, spesso indicato come **ERD**.

La **progettazione concettuale** può essere rappresentata in questo modo:

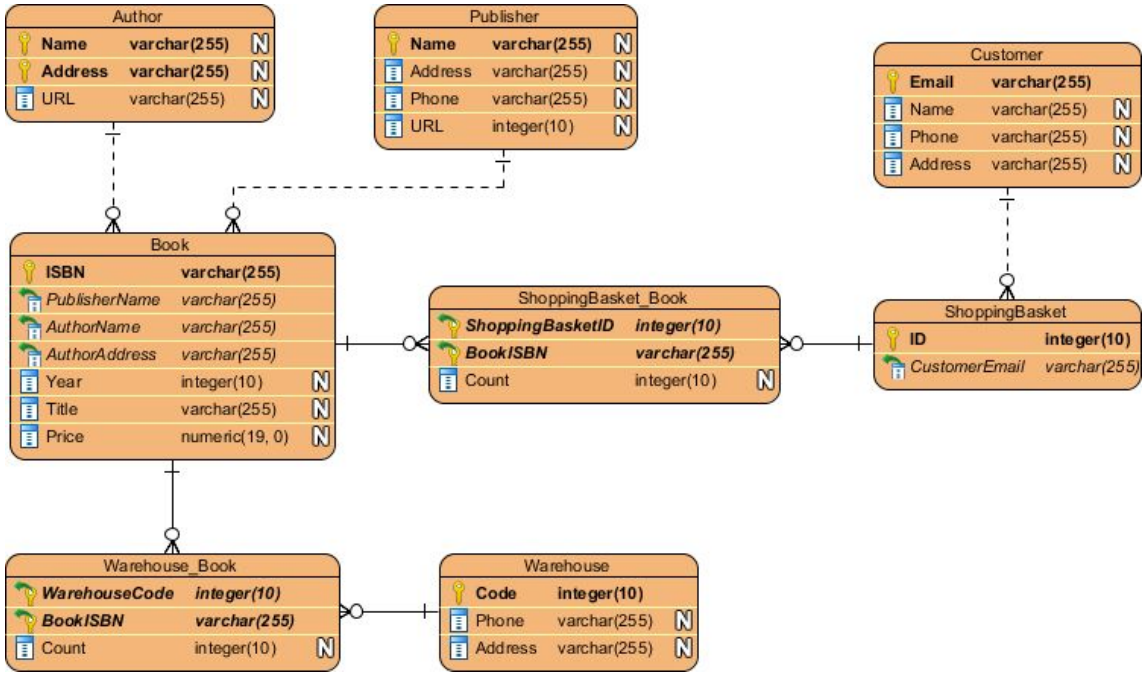


A cui possiamo integrare la **progettazione logica:**

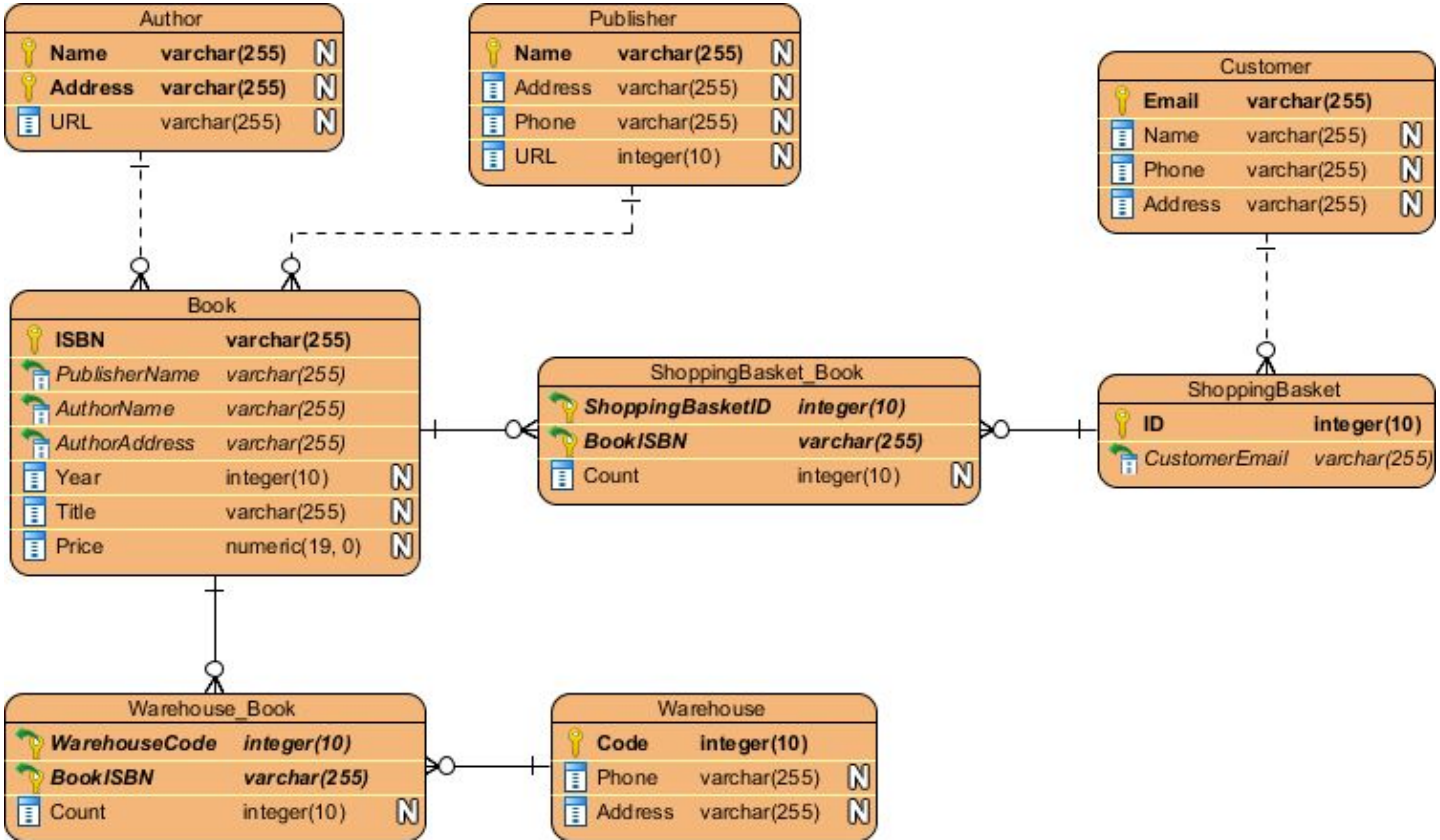


Progettazione Fisica

Nel momento in cui integriamo l'ultima fase, la **progettazione fisica**, ci ritroviamo con un **ERD** completo. Sviluppando gli esempi precedenti avremo:

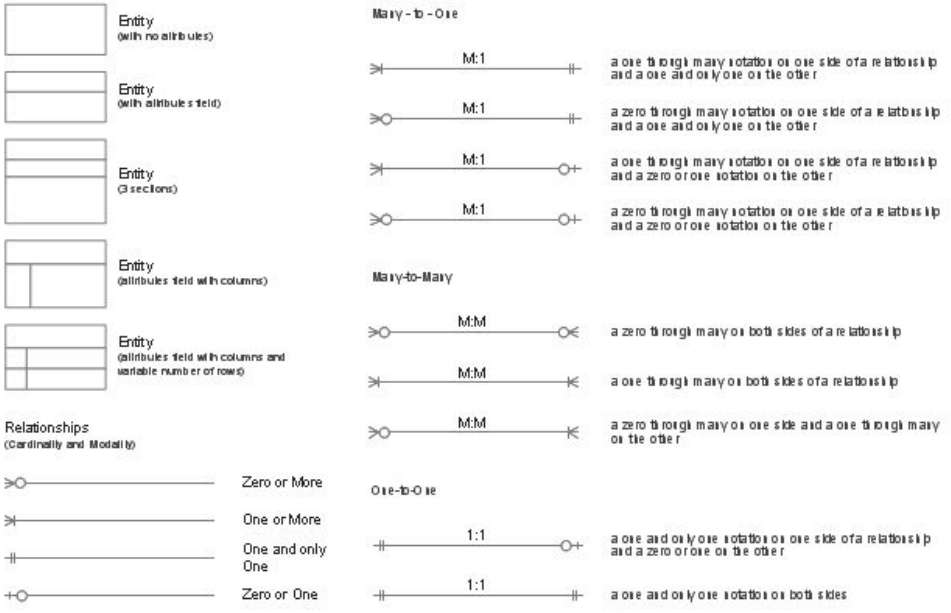


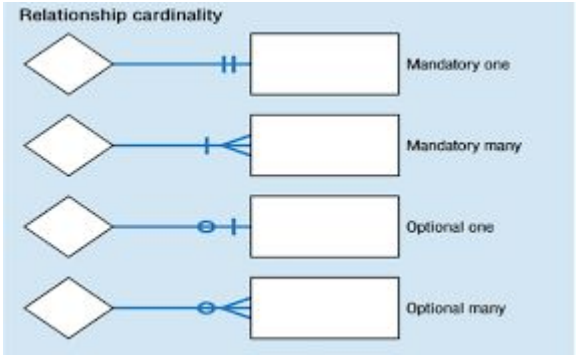
Progettazione Fisica - Schema (ingrandimento)



Rappresentazioni

Le linee di collegamento tra le tabelle rappresentano sia le relazioni che la cardinalità.
Teniamo presente **che non c'è sempre univocità sulle rappresentazioni grafiche**: per questo motivo è buona pratica inserire una **legenda a margine del nostro ERD**, come una delle seguenti:





Conclusioni

Ma in quali casi ci può essere utile disegnare un **ERD**, oltre all'implementazione iniziale?

In molti, alcuni esempi possono essere:

- **Aiuto nella raccolta dei requisiti** - Determinare i requisiti di un sistema informativo disegnando un ERD concettuale che rappresenta gli oggetti aziendali di alto livello del sistema. Un tale modello iniziale può anche evolversi in un modello di database fisico che facilita la creazione di un database relazionale o aiuta nella creazione di mappe di processo e modelli di flusso di dati.

- **Progettazione delle modifiche al database** - A seconda della portata del cambiamento, può essere rischioso modificare direttamente la struttura di un database in un DBMS. Per evitare di compromettere i dati in un database di produzione, è importante pianificare attentamente le modifiche. Disegnando diagrammi ER per visualizzare le idee di progettazione del database, si ha la possibilità di individuare gli errori e i difetti di progettazione e di apportare correzioni prima di eseguire le modifiche nel database.

- **Debugging del database** - Risolvere i problemi di un database può essere impegnativo, specialmente quando il database contiene molte tabelle, il che richiede la scrittura di query complesse per ottenere le informazioni necessarie. Visualizzando uno schema di database con un ERD, si ha una visione d'insieme più precisa, che consente di analizzare un database esistente e di individuarne più facilmente i problemi.

EPICODE

GRAZIE

Epicode