UNIVERSITÀ
DEGLI STUDI
DI PADOVA

DISORDER STORE DB

Progetto di basi di dati
Anno 2016/2017

Riccardo Patanè *1125514*
Anna Poletti *1123587*

ABSTRACT

Il negozio di dischi “Disorder Store”, per festeggiare i 50 anni di attività, decide di rendere disponibile su una piattaforma online tutto il proprio archivio in modo che ogni cliente possa collegarsi e ascoltare i propri brani preferiti.
Il catalogo del negozio è formato sia da album sia da singoli appartenenti a vari generi e artisti. Oltre che ad ascoltare i brani presenti nell’archivio, il cliente può creare delle playlist, ovvero delle liste di canzoni che permettono la gestione più rapida dei brani in esecuzione e la loro sequenza. I clienti possono collaborare tra di loro e creare playlist condivise che contengono le canzoni scelte da loro.
Il database “Disorder Store DB” gestisce le funzioni elencate sopra e si propone di essere una piattaforma con delle funzioni fondamentali: catalogazione della musica, ricerca per parole chiave, creazione e modifica di playlist da parte degli utenti, classifica delle canzoni e dati statistici.

DESCRIZIONE

Si vuole realizzare una base di dati che contenga e gestisca le informazioni relative all’archivio del negozio di dischi “Disorder Store”.
Ogni brano si divide in singolo o traccia e di entrambe le tipologie si vogliono sapere:
- il titolo della canzone
- la durata espressa come orario
- l’artista che suona o canta il pezzo
- se è adatto o no a un pubblico di minori.
- il genere a cui appartiene, del quale vogliamo conoscere il nome, l’anno e la nazione di nascita
- l’anno di uscita
-l’etichetta discografica produttrice, della quale vogliamo sapere nome e se è indipendente o non lo è.

Dell’artista, che può essere solista o un gruppo musicale, vogliamo conoscere:
- il nome d’arte
- l’anno di inizio
- l’anno fine carriera nel caso l’artista non fosse più in attività
- gli strumenti suonati, se presenti.
- il numero di componenti solo se l’artista è una band

Una traccia appartiene ad un album, del quale si vuole sapere:
- titolo dell’album
- anno di uscita
- numero di tracce

Del cliente del negozio, identificato univocamente dal codice della carta fedeltà posseduta, vogliamo conoscere:
- nome e cognome
- anno di nascita
- sesso
- nazione di residenza, di cui vogliamo conoscere il nome
- le playlist create e quelle in cui collabora

Delle playlist vogliamo conoscere:
-il titolo
-il numero di tracce

Il database tiene in memoria il primo ascolto della giornata di un certo brano da parte di un cliente.

GLOSSARIO DEI TERMINI

Termine	Descrizione	Sinonimi	Collegamenti
Canzone	Composizione vocale con accompagnamento strumentale	Brano, pezzo	Cliente, playlist
Singolo	Brano che non appartiene a nessun album		Etichetta, genere, artista
Traccia	Brano che appartiene a un album		Album
Artista	Colui che canta o suona un brano		Strumento, album
Solista	Artista che non fa parte di un gruppo musicale	Cantante	
Band	Gruppo di persone che suona e canta	Gruppo musicale	
Strumento	Oggetto che produce suoni	Strumento musicale	Artista
Etichetta	Compagnia che produce e distribuisce un singolo o un album	Etichetta discografica, casa discografica	Album, singolo
Ascolto	Riproduzione multimediale di una canzone		Cliente, canzone
Cliente	Persona autorizzata ad accedere al database	Utente	Carta fedeltà, canzone, playlist, nazione
Carta fedeltà	Codice identificativo del cliente	Tessera, id	Cliente
Playlist	Lista di canzoni		Cliente
Nazione		Paese, stato	Cliente
Genere	Categoria di classificazioni dei brani a seconda di affinità		Singolo, album
Album	Raccolta di brani legati ad un artista	Disco	Artista, traccia, genere, etichetta

LISTA DELLE CLASSI

Canzone: singolo o traccia
- titolo: varchar(20), not null
- durata: time, not null
- esplicito: booleano
- id: int, not null, chiave

Singolo:
-anno_uscita:int(4)

Traccia:
-numero_traccia: int(2)

Genere:
-nome: varchar(20), not null, chiave
-nazione: varchar(20)
- anno_nascita: int(4)

Album:
-titolo: varchar(20), not null
-anno_uscita: int(4)
-numero_tracce: int(2)

Artista: solista o band:
-nome_d_arte: varchar(20), not null, chiave
-anno_inizio_carriera: int(4), not null
-anno_fine_carriera: int(4)

Band:
-numero_componenti: int(2)

Strumento:
-nome: varchar(20), not null, chiave

Etichetta:
-nome: varchar(20), not null, chiave
-indipendente booleano

Cliente:
-nome: varchar(20), not null
-cognome: varchar(20), not null
-sesso: varchar(10)
-anno_nascita: int(4)
Carta Fedeltà è chiave esterna

Carta fedeltà:
-numero:int(5), not null, chiave

Playlist:
-titolo: varchar(20), not null
-numero_tracce: int(3)
Titolo, Carta fedeltà è chiave esterna

Nazione:
-nome: varchar(20), not null, chiave

LISTA DELLE RELAZIONI

Suonare: collega artista a strumento. Un artista può suonare da 0 a N strumenti, uno strumento può essere suonato da 0 a N artisti.

Appartenere: collega sia singolo sia album a genere. Un singolo appartiene ad un genere, un genere può raccogliere da 0 a N singoli.
Un album appartiene ad un genere, un genere può raccogliere da 0 a N album.

Produrre: collega sia a singolo sia album a etichetta. Un singolo può essere prodotto da una sola etichetta, un etichetta può produrre da 0 a N singoli.
Un album può essere prodotto da una sola etichetta, un etichetta può produrre da 0 a N album.

Contiene: collega traccia a album. Una traccia può essere contenuta in un solo album, un album può contenere da 1 a N tracce.

Incide: collega sia album sia singolo ad artista. Un singolo può essere inciso da un solo artista, mentre un artista può incidere da 0 a N singoli.
Un album può essere inciso da un solo artista, mentre un artista può incidere da 0 a N album.

Ascolto: collega canzone a cliente. Una canzone può essere ascoltata da 0 a N clienti, un cliente può ascoltare da 0 a N canzoni.
-data: date

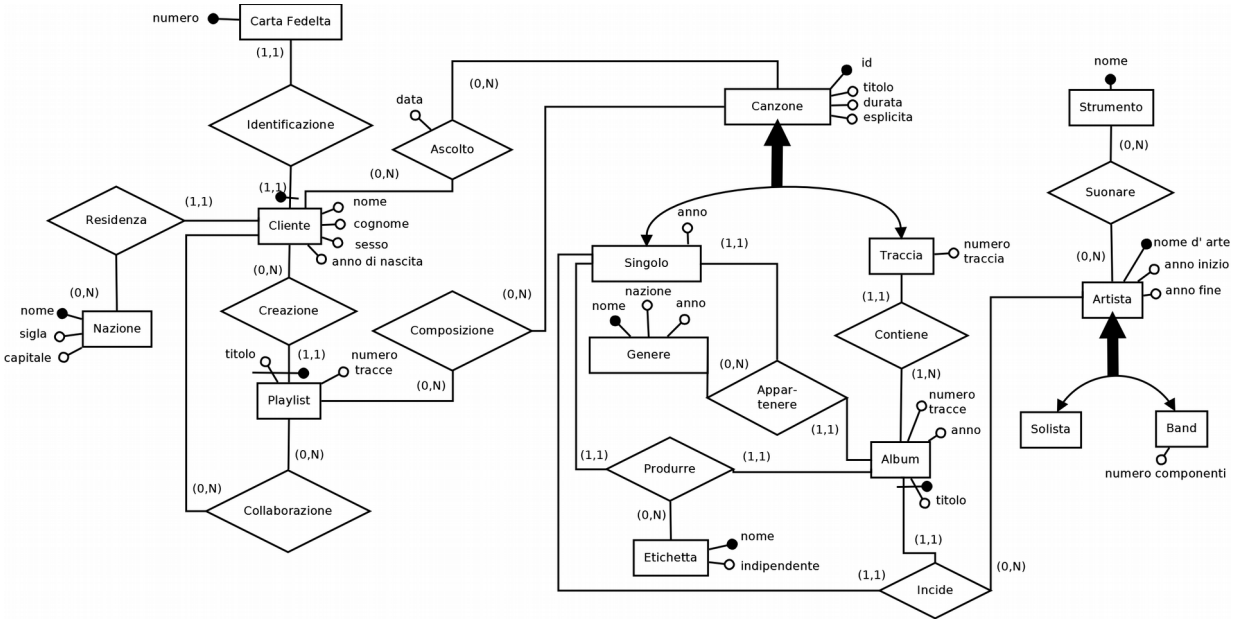
Composizione: collega playlist a cliente. Una playlist può essere composta da 0 a N canzoni. Una canzone può essere da 0 a N playlist.

Identificazione: collega cliente a carta fedeltà. Un cliente è identificato da una sola carta fedeltà, una carta fedeltà identifica un solo cliente.

Residenza: collega nazione a cliente. In una nazione possono risiedere da 0 a N clienti, un cliente può risiedere in una sola nazione.

Collaborazione: collega cliente a playlist. Un cliente può collaborare da 0 a N playlist, una playlist può avere da 0 a N clienti che collaborano.

Creazione: collega cliente a playlist. Un cliente può creareda 0 a N playlist, una playlist può essere creata da un solo cliente



PROGETTAZIONE LOGICA

Ristrutturazione dello schema

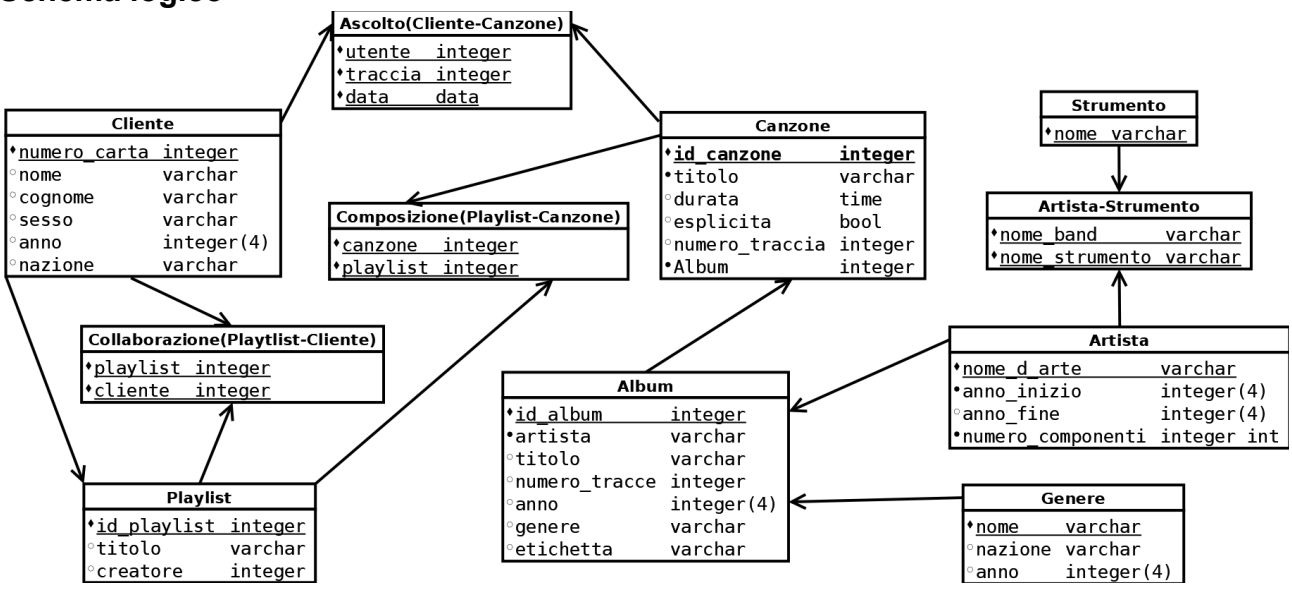
-Analisi delle ridondanze:
Vi è una ridondanza tra l'attributo *Durata* dell'entità *Album* e l'attributo *Durata* dell'entità C*anzone*. Risolviamo questa ridondanza togliendo l'attributo *Durata* dall'entità *Album* poiché possiamo ricavarla attraverso una query.
È presente un' altra ridondanza nell'attributo *Numero Tracce* dell'entità *Album.* Possiamo togliere questo attributo poiché possiamo ricavarlo dal valore più alto che assume l'attributo *Numero traccia* dell'entità *Traccia.*

-Eliminazione delle generalizzazioni:
Nello schema ER sono presenti due generalizzazioni totali: una che ha come entità madre *Canzone* e come entità figlie *Singolo* e *Traccia,* l'altra che ha come entità madre *Artista* e come entità figlie *Solista* e *Band*.
Entrambe vengono tradotte accorpamento delle figlie della generalizzazione nel genitore.
In pratica all'entità *Canzone* viene aggiunto l'attributo *Numero Traccia* e sarà collegata dalla relazione *Contiene* ad *Album*. Se l'attributo *Titolo* dell'entità *Album* è uguale all'attributo *Titolo* dell'entità *Canzone* e contemporaneamente l'attributo *Numero Tracce* dell'entità Album è uguale a 1, la canzone che appartiene all'album è un singolo.
All'entità *Artista* viene aggiunto l'attributo *Numero Componenti:* se il numero dei componenti è uguale a 1, l'artista è un solista.

-Partizionamento/accorpamento di entità e relationship:
Poiché hanno una relazione con entrambe le cardinalità uno a uno, è possibile accorpare l'entità Carta fedeltà all'entità Cliente e farla diventare un attributo.
L'entità *Nazione* può essere accorpata all'entità *Cliente* e diventare attributo.
L'entità *Etichetta* possiede due attributi (*Nome* e *Indipendente*) e poiché l'attributo *Indipendente* è superfluo,è possibile eliminarlo e far diventare l'entità *Etichetta* un attributo dell'entità *Album*.

-Scelta degli identificatori primari:
Non tutte le entità possono essere identificate univocamente attraverso un solo attributo, tra queste vi sono le entità *Album e Playlist.*
Inseriremo quindi un ID numerico per ognuno di esse per identificarle.

Schema logico



Schema relazionale

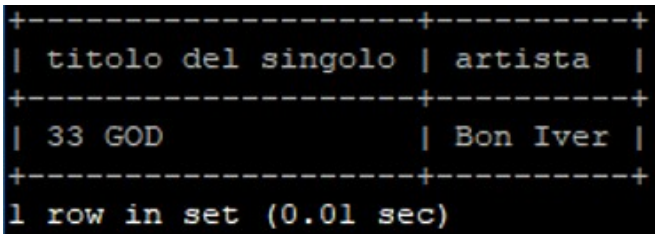
- Canzone(**id_canzone,** titolo, durata, esplicita, numero_traccia, album)
- Artista(**nome_d_arte**, anno_inizio, anno_fine, numero_componenti)
- Strumento(**nome**)
- Artista_Strumento(**nome_band, nome_strumento**)
- Genere(**nome**, nazione, anno)
- Album(**id_album**, artista, titolo, numero_tracce, anno, genere, etichetta)
- Cliente(**numero_carta**, nome, cognome, sesso, anno, nazione)
- Ascolto(**utente, traccia, data**)
- Playlist(i**d_playlist**, titolo, creatore)
- Collaborazione(**playlist, cliente**)
- Composizione(**canzone, playlist**)

QUERY, TRIGGER E FUNZIONI

QUERY

1. Mostrare i nomi di tutti i singoli appartenenti agli artisti ancora in attività.
(Un singolo è un brano appartenente ad un album omonimo che contiene una sola traccia.)

```
SELECT c.titolo as "titolo del singolo",ar.nome_d_arte as "artista "
FROM Canzone as c JOIN Album as al on c.album=al.id_album
JOIN Artista as ar on al.artista=ar.nome_d_arte
WHERE c.numero_traccia=1 AND al.numero_tracce=1 AND c.titolo=al.titolo AND
ar.anno_fine IS NULL;
```



```
+--------------------+----------+
| titolo del singolo | artista  |
+--------------------+----------+
| 33 GOD             | Bon Iver |
+--------------------+----------+
1 row in set (0.01 sec)
```

2. Mostrare i clienti che ascoltano almeno una canzone di
un genere nato nella loro stessa nazione.

```
SELECT DISTINCT cl.nome, cl.cognome, cl.nazione
FROM Cliente as cl JOIN Ascolto as asco on cl.numero_carta=asco.utente
JOIN Canzone as ca on ca.id_canzone=asco.traccia
JOIN Album as al on ca.album=al.id_album JOIN Genere as ge on al.genere=ge.nome
WHERE ge.nazione=cl.nazione;
```

```
+--------+------------+-------------+
| nome   | cognome    | nazione     |
+--------+------------+-------------+
| Brian  | Warner     | USA         |
| Stanis | LaRochelle | Inghilterra |
+--------+------------+-------------+
2 rows in set (0.00 sec)
```

3. Mostrare i minuti di utilizzo della piattaforma di ogni cliente in ordine decrescente

```
SELECT cl.nome, cl.cognome, sum(durata)/60 as minuti_di_ascolto
FROM Cliente as cl JOIN Ascolto as asco on cl.numero_carta=utente
JOIN Canzone on traccia=id_canzone
GROUP BY cl.numero_carta
ORDER BY minuti_di_ascolto DESC;
```

```
+---------+-----------+-------------------+
| nome    | cognome   | minuti_di_ascolto |
+---------+-----------+-------------------+
| Brian   | Warner    |           40.7833 |
| Stanis  | LaRochelle|           38.2333 |
| Alba    | Gobbo     |           36.2667 |
| Maria   | Boldo     |           35.5500 |
| Beto    | Rea       |           26.1167 |
| Etienne | Pixa      |           22.0333 |
| Gloria  | Delgado   |           13.7167 |
+---------+-----------+-------------------+
7 rows in set (0.00 sec)
```

4. Si vuole conoscere l'artista che suona più strumenti

```
SELECT a.nome_d_arte, COUNT(a.nome_d_arte) as numero_strumenti
FROM Artista as a JOIN Artista_Strumento as sa on a.nome_d_arte=sa.nome_band
JOIN Strumento as s on sa.nome_strumento=s.nome
GROUP BY nome_d_arte
HAVING numero_strumenti >= ALL (SELECT COUNT(a.nome_d_arte)
                                FROM Artista as a JOIN Artista_Strumento as sa on
                                a.nome_d_arte=sa.nome_band
                                JOIN Strumento as s on sa.nome_strumento=s.nome
                                GROUP BY nome_d_arte);
```

```
+-------------+------------------+
| nome_d_arte | numero_strumenti |
+-------------+------------------+
| Bon Iver    |                4 |
+-------------+------------------+
1 row in set (0.00 sec)
```

5. Mostrare gli utenti che non possiedono e non collaborano a nessuna playlist

```
SELECT nome,cognome
FROM Cliente
WHERE numero_carta != ALL (SELECT cliente FROM Collaborazione) AND
numero_carta != ALL(SELECT creatore FROM Playlist);
```

nome	cognome
Brian	Warner
Beto	Rea

2 rows in set (0.00 sec)

6. Mostrare tutte canzoni che sono state ascoltate almeno una volta ordinate per numero di ascoltatori

```
SELECT Canzone.titolo,nome_d_arte,Count(Ascolto.utente) as NumeroAscoltatori
FROM Canzone JOIN Album on album=id_album
JOIN Artista on artista=nome_d_arte JOIN Ascolto on id_canzone=traccia
GROUP BY Ascolto.utente,Ascolto.traccia
ORDER BY NumeroAscoltatori DESC;
```

titolo	nome_d_arte	NumeroAscoltatori
Made of Metal	Grouper	3
Sint Hack	Nick Cave	2
Titan Arch	Coil	2
Cabin Fever	Nick Cave	1
Jesus Alone	Nick Cave	1
Made of Metal	Grouper	1
Testa di Cane	Zu	1
Dark River	Coil	1
Titan Arch	Coil	1
Made of Air	Grouper	1
Clearing	Grouper	1
Holocene	Bon Iver	1
From Her to Eternity	Nick Cave	1
Anthrocene	Nick Cave	1
Detonatore	Zu	1
Titan Arch	Coil	1
The Snow	Coil	1
Holding	Grouper	1
Michicant	Bon Iver	1
Holocene	Bon Iver	1
From Her to Eternity	Nick Cave	1
Cabin Fever	Nick Cave	1
Jesus Alone	Nick Cave	1
Testa di Cane	Zu	1
Dark River	Coil	1
The Snow	Coil	1
Holding	Grouper	1
Holocene	Bon Iver	1
Sint Hack	Nick Cave	1
From Her to Eternity	Nick Cave	1

30 rows in set (0.00 sec)

TRIGGER

1.Trigger che ad ogni ad ogni nuova canzone inserita, aggiorna il numero di tracce dell'album.

```
DROP TRIGGER IF EXISTS insertCanzone;
DELIMITER |
CREATE TRIGGER insertCanzone AFTER INSERT ON Canzone
FOR EACH ROW
BEGIN

UPDATE Album SET numero_tracce=(SELECT COUNT(id_canzone) FROM
Canzone WHERE album=id_album) ;

END|
DELIMITER ;
```

2. Trigger che ad ogni inserimento riguardante la collaborazione di un utente a una playlist, verifica che il nuovo utente collaboratore non sia anche l'utente creatore della playlist.
Nel caso in cui l'utente inserito nella collaborazione della playlist fosse anche il creatore, il trigger genera un messaggio di errore.

```
DROP TRIGGER IF EXISTS NuovaCollabo;
DELIMITER |
CREATE TRIGGER NuovaCollabo
BEFORE INSERT on Collaborazione
FOR EACH ROW

BEGIN

IF(new.cliente=(SELECT creatore
FROM Playlist
WHERE new.playlist=id_playlist))
THEN
SIGNAL SQLSTATE '45000'
SET MESSAGE_TEXT = 'ERRORE il nuovo collaboratore e anche il creatore';

END IF;

END|

DELIMITER ;
```

FUNZIONI

1. La funzione permette di ricavare il numero di ascolti per ogni utente. Dato il numero della tessera fedeltà e la funzione restituisce di ascolti.

```
DROP FUNCTION IF EXISTS NumeroAscolti;
DELIMITER |
CREATE FUNCTION NumeroAscolti (numTessera INTEGER) RETURNS INTEGER
BEGIN

DECLARE Ascolti INT;

SELECT COUNT (utente)
FROM Ascolto
WHERE utente=numTessera INTO Ascolti;

RETURN Ascolti;
END|

DELIMITER ;
```

2. La funzione permette di ricavare il numero delle volte in cui le tracce di un album sono state ascoltate
Quindi, dato il codice identificativo dell'album, la funzione restituisce la somma degli ascolti di tutte le canzoni appartenenti all'album.

```
DROP FUNCTION IF EXISTS NumeroAscoltiAlbum;

DELIMITER |

CREATE FUNCTION NumeroAscoltiAlbum (AlbumID INTEGER)
RETURNS INTEGER
BEGIN

DECLARE Ascolti=0 INT;

SELECT COUNT(Ascolto.traccia)
FROM Ascolto JOIN Canzone ON Ascolto.traccia=Canzone.id_canzone
JOIN Album ON Canzone.Album=Album.id_album
WHERE id_album=AlbumID INTO Ascolti;

RETURN Ascolti;
END|

DELIMITER ;
```