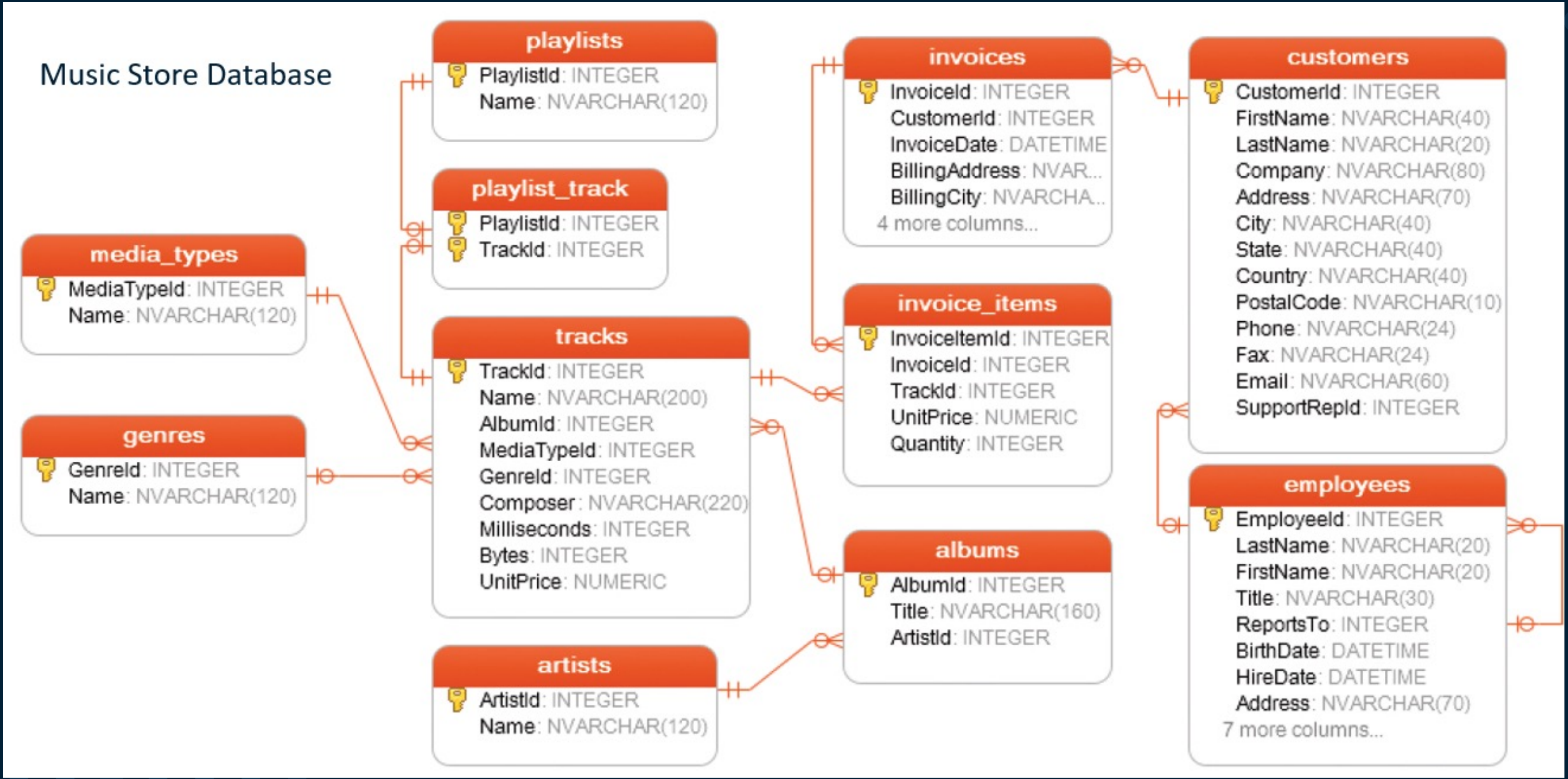
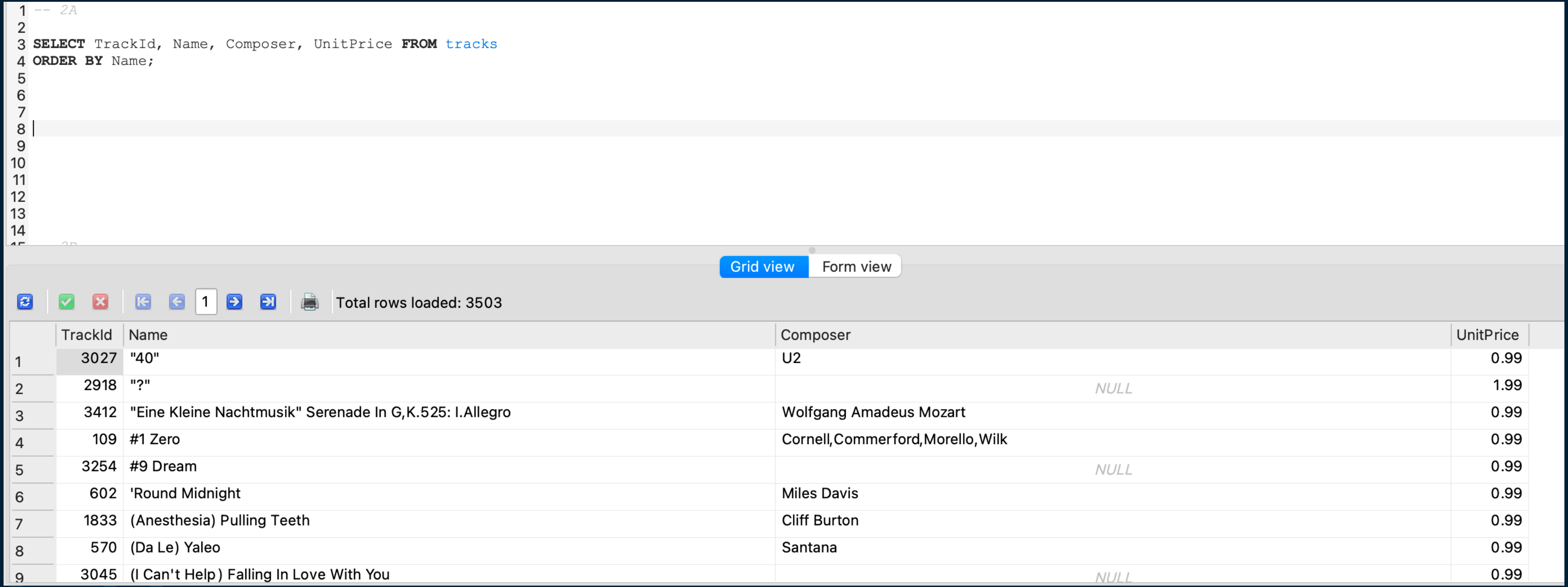
Data Management

Online Music Store Data Model



SQL Queries

Create a report showing the track id, name, composer, and unit price of all of the tracks that are sold by the company sorted by name:



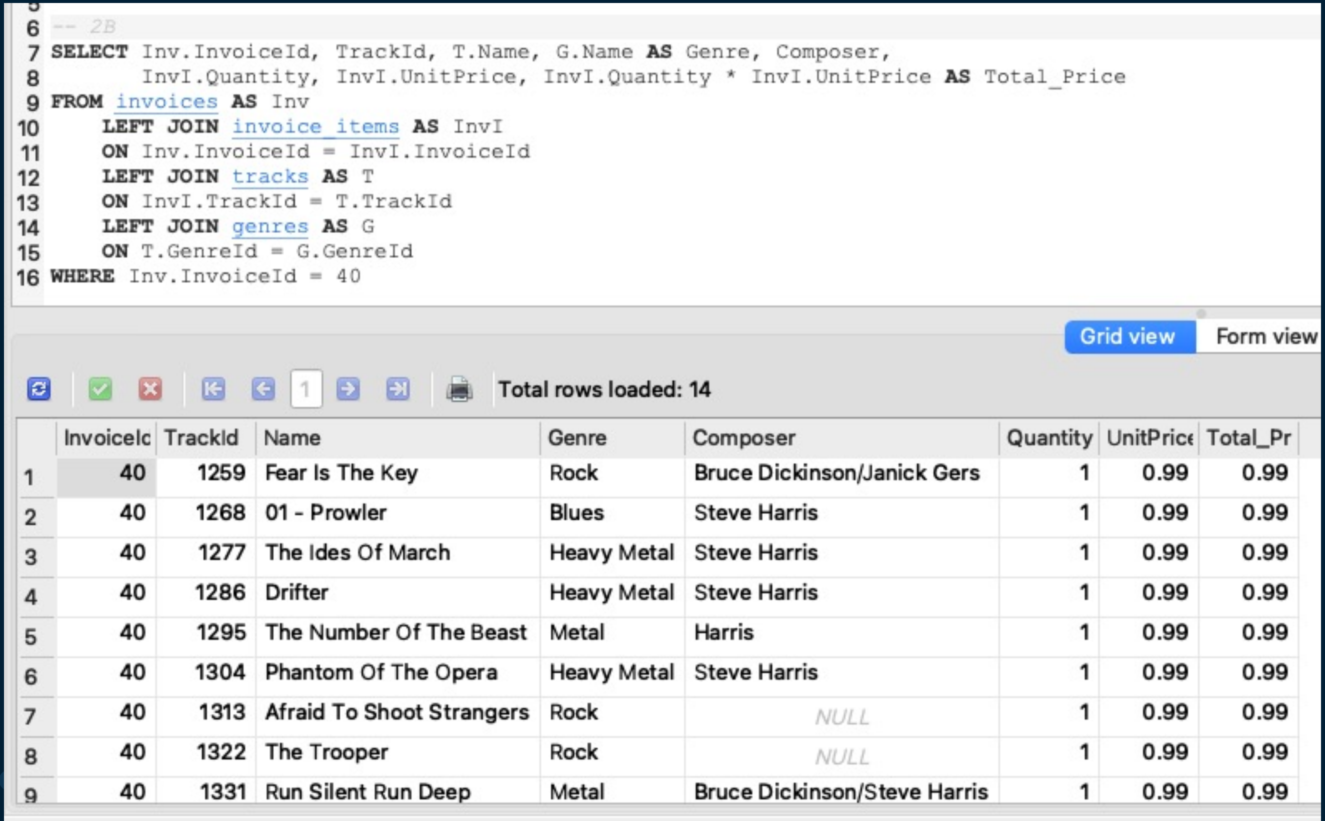
```
-- 2A

SELECT TrackId, Name, Composer, UnitPrice FROM tracks
ORDER BY Name;
```

Total rows loaded: 3503

	TrackId	Name	Composer	UnitPrice
1	3027	"40"	U2	0.99
2	2918	"?"	NULL	1.99
3	3412	"Eine Kleine Nachtmusik" Serenade In G,K.525: I.Allegro	Wolfgang Amadeus Mozart	0.99
4	109	#1 Zero	Cornell,Commerford,Morello,Wilk	0.99
5	3254	#9 Dream	NULL	0.99
6	602	'Round Midnight	Miles Davis	0.99
7	1833	(Anesthesia) Pulling Teeth	Cliff Burton	0.99
8	570	(Da Le) Yaleo	Santana	0.99
9	3045	(I Can't Help) Falling In Love With You	NULL	0.99

SQL Queries

Create a report for invoice #40 showing the TrackID, Name, Genre, Composer, Quantity, Unit Price, and Total Price for each track on that invoice.



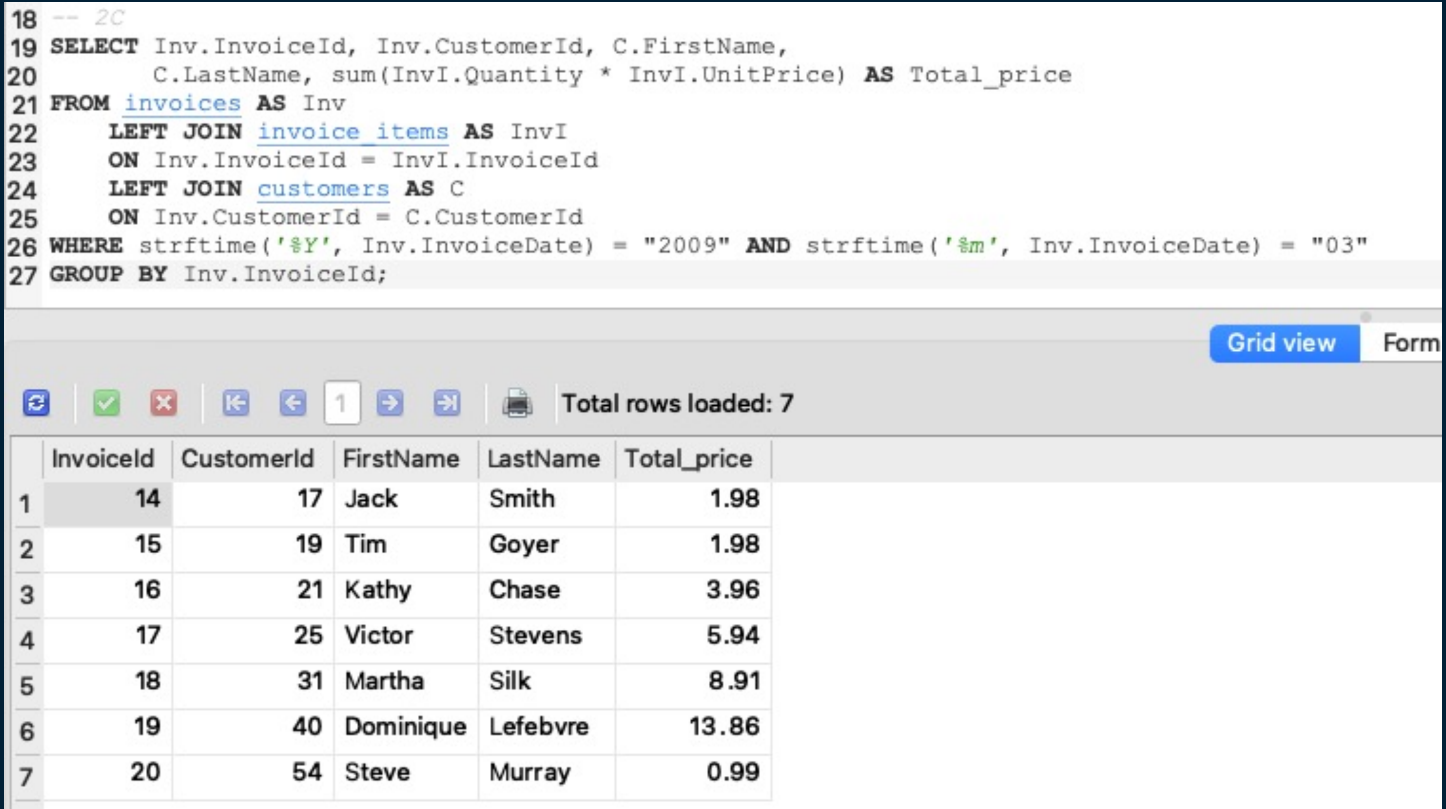
```
-- 2B
SELECT Inv.InvoiceId, TrackId, T.Name, G.Name AS Genre, Composer,
       InvI.Quantity, InvI.UnitPrice, InvI.Quantity * InvI.UnitPrice AS Total_Price
FROM invoices AS Inv
    LEFT JOIN invoice_items AS InvI
    ON Inv.InvoiceId = InvI.InvoiceId
    LEFT JOIN tracks AS T
    ON InvI.TrackId = T.TrackId
    LEFT JOIN genres AS G
    ON T.GenreId = G.GenreId
WHERE Inv.InvoiceId = 40
```

Total rows loaded: 14

	InvoiceIc	TrackId	Name	Genre	Composer	Quantity	UnitPrice	Total_Pr
1	40	1259	Fear Is The Key	Rock	Bruce Dickinson/Janick Gers	1	0.99	0.99
2	40	1268	01 - Prowler	Blues	Steve Harris	1	0.99	0.99
3	40	1277	The Ides Of March	Heavy Metal	Steve Harris	1	0.99	0.99
4	40	1286	Drifter	Heavy Metal	Steve Harris	1	0.99	0.99
5	40	1295	The Number Of The Beast	Metal	Harris	1	0.99	0.99
6	40	1304	Phantom Of The Opera	Heavy Metal	Steve Harris	1	0.99	0.99
7	40	1313	Afraid To Shoot Strangers	Rock	NULL	1	0.99	0.99
8	40	1322	The Trooper	Rock	NULL	1	0.99	0.99
9	40	1331	Run Silent Run Deep	Metal	Bruce Dickinson/Steve Harris	1	0.99	0.99

SQL Queries

Create a report listing each invoice issued in March, 2009 and showing the Invoice ID, Customer ID, Customer First and Last Name, and Total Price



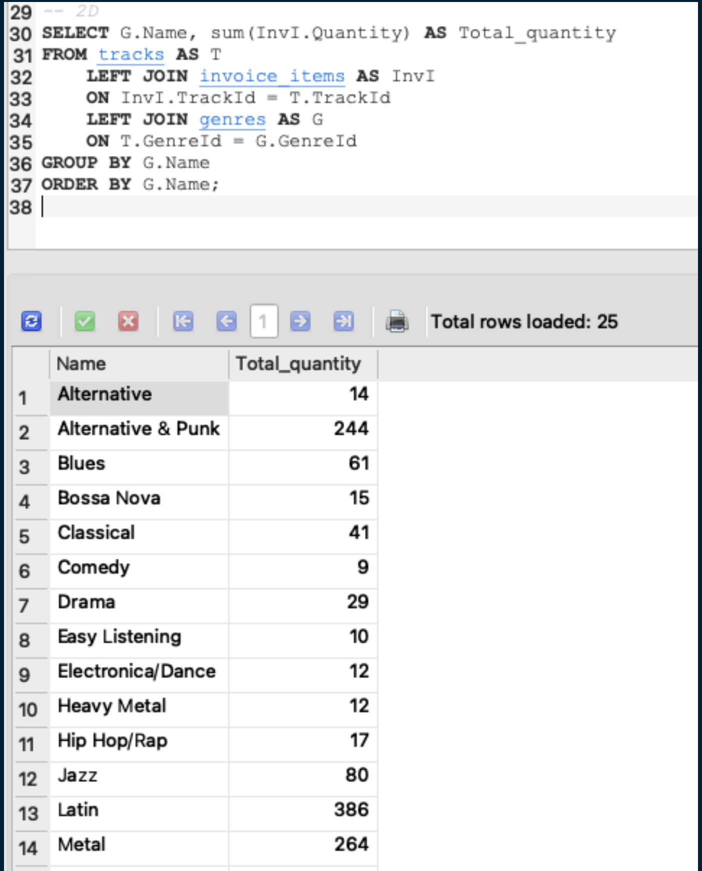
```
-- 2C
SELECT Inv.InvoiceId, Inv.CustomerId, C.FirstName,
       C.LastName, sum(InvI.Quantity * InvI.UnitPrice) AS Total_price
FROM invoices AS Inv
    LEFT JOIN invoice_items AS InvI
    ON Inv.InvoiceId = InvI.InvoiceId
    LEFT JOIN customers AS C
    ON Inv.CustomerId = C.CustomerId
WHERE strftime('%Y', Inv.InvoiceDate) = "2009" AND strftime('%m', Inv.InvoiceDate) = "03"
GROUP BY Inv.InvoiceId;
```

Total rows loaded: 7

	InvoiceId	CustomerId	FirstName	LastName	Total_price
1	14	17	Jack	Smith	1.98
2	15	19	Tim	Goyer	1.98
3	16	21	Kathy	Chase	3.96
4	17	25	Victor	Stevens	5.94
5	18	31	Martha	Silk	8.91
6	19	40	Dominique	Lefebvre	13.86
7	20	54	Steve	Murray	0.99

SQL Queries

Create a report showing the total quantity of tracks sold for each category sorted by category name.



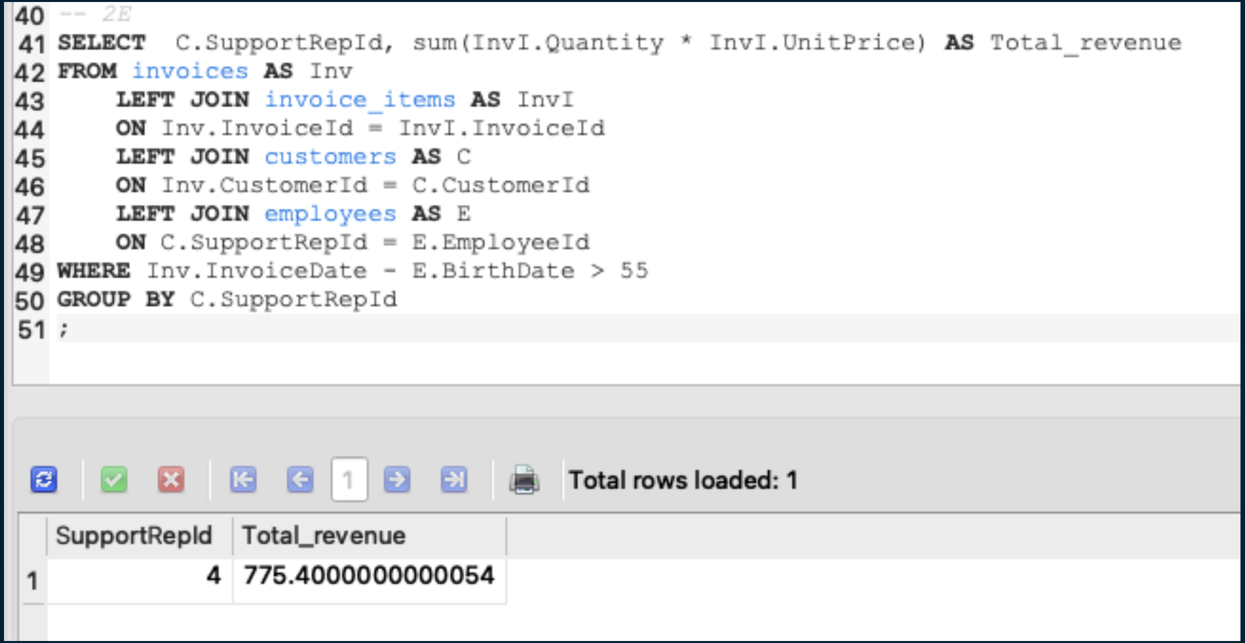
```
-- 2D
SELECT G.Name, sum(InvI.Quantity) AS Total_quantity
FROM tracks AS T
    LEFT JOIN invoice_items AS InvI
    ON InvI.TrackId = T.TrackId
    LEFT JOIN genres AS G
    ON T.GenreId = G.GenreId
GROUP BY G.Name
ORDER BY G.Name;
```

Total rows loaded: 25

	Name	Total_quantity
1	Alternative	14
2	Alternative & Punk	244
3	Blues	61
4	Bossa Nova	15
5	Classical	41
6	Comedy	9
7	Drama	29
8	Easy Listening	10
9	Electronica/Dance	12
10	Heavy Metal	12
11	Hip Hop/Rap	17
12	Jazz	80
13	Latin	386
14	Metal	264

SQL Queries

Create a report showing the total revenue associated with each support rep who is over 55 years old (as of the date of running the report)



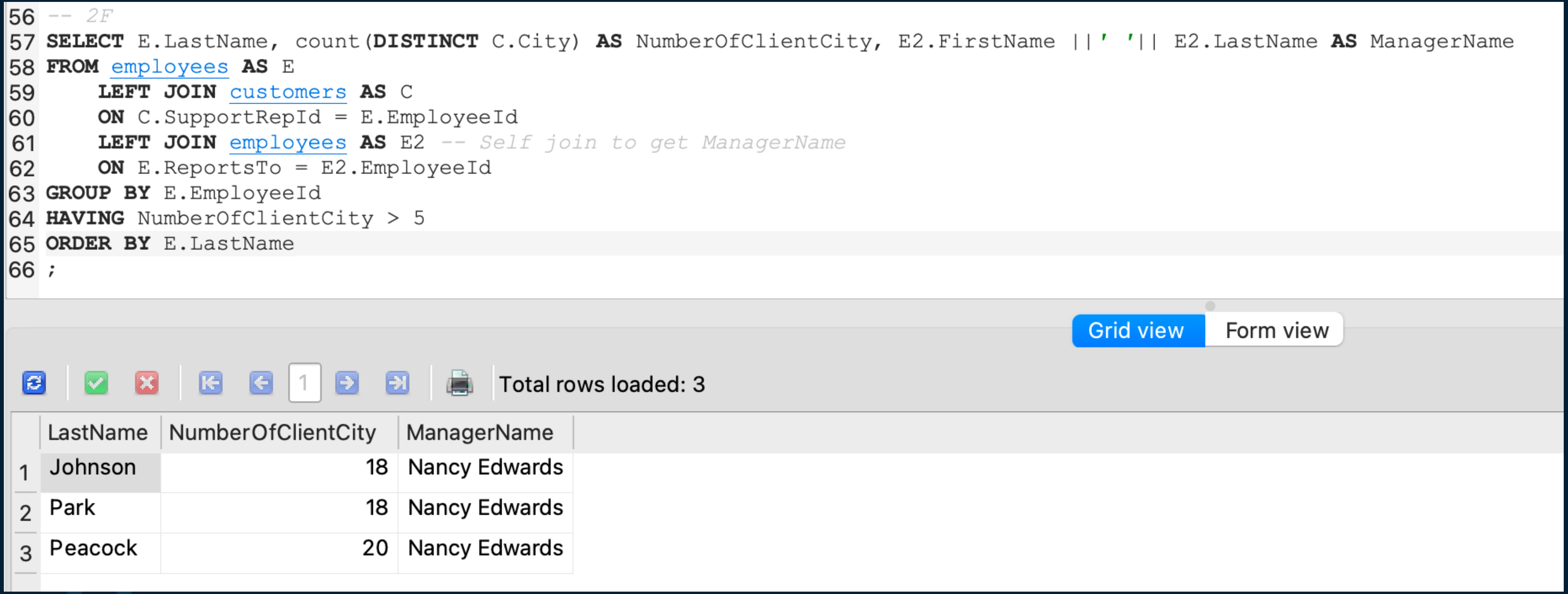
```
-- 2E
SELECT  C.SupportRepId, sum(InvI.Quantity * InvI.UnitPrice) AS Total_revenue
FROM invoices AS Inv
    LEFT JOIN invoice_items AS InvI
    ON Inv.InvoiceId = InvI.InvoiceId
    LEFT JOIN customers AS C
    ON Inv.CustomerId = C.CustomerId
    LEFT JOIN employees AS E
    ON C.SupportRepId = E.EmployeeId
WHERE Inv.InvoiceDate - E.BirthDate > 55
GROUP BY C.SupportRepId
;
```

Total rows loaded: 1

	SupportRepId	Total_revenue
1	4	775.4000000000054

SQL Queries

Write a query to return the last name of all of employees who serve customers from more than 5 different cities along with the name of the manager that they report to. Sort by the last name of the employee.



```
-- 2F
SELECT E.LastName, count(DISTINCT C.City) AS NumberOfClientCity, E2.FirstName ||' '|| E2.LastName AS ManagerName
FROM employees AS E
    LEFT JOIN customers AS C
    ON C.SupportRepId = E.EmployeeId
    LEFT JOIN employees AS E2 -- Self join to get ManagerName
    ON E.ReportsTo = E2.EmployeeId
GROUP BY E.EmployeeId
HAVING NumberOfClientCity > 5
ORDER BY E.LastName
;
```

Grid view | Form view

Total rows loaded: 3

	LastName	NumberOfClientCity	ManagerName
1	Johnson	18	Nancy Edwards
2	Park	18	Nancy Edwards
3	Peacock	20	Nancy Edwards

Data Warehouse Design

Sales Datamart

You now wish to design a sales DataMart for the online music store. Your sales business users wish to be able to ask questions such as:

- What was the total revenue by track for a specific day (Monday, Tuesday, …)/month (1-12)/ week (1-52) /year?
  - For a specific album? For a specific genre?
- What was the total revenue by customer city? country?
- What was the total revenue by media type?

Data Warehouse Design

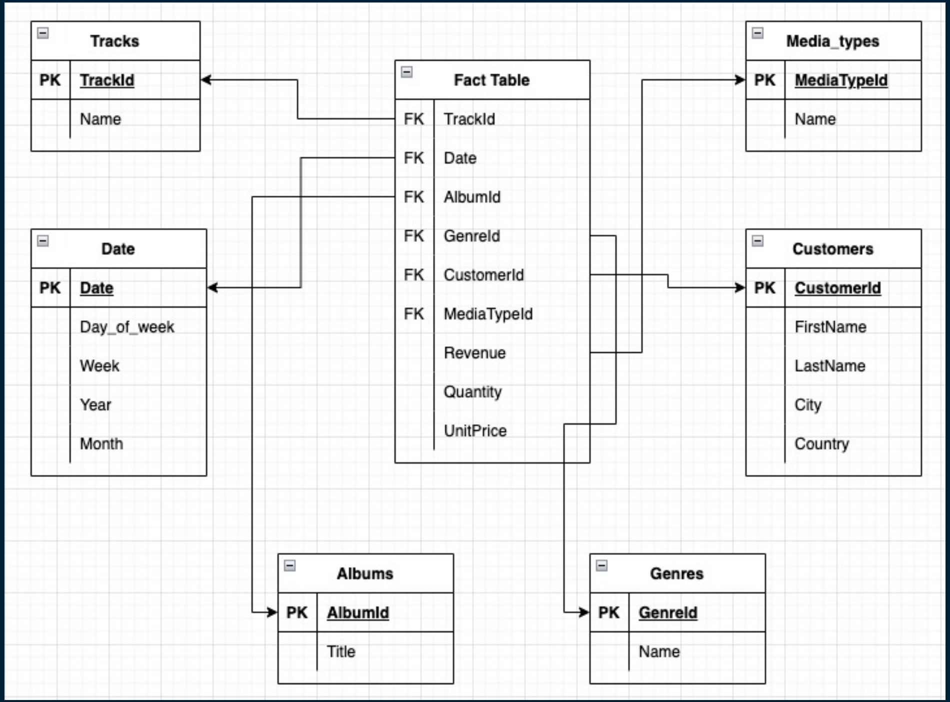
Sales Datamart

- What is the grain? **One row represents a track on a specific date, an album, a genre, a customer, and a media type**
- What are the dimensions? **Tracks, Date, Album, Genre, Customer, Media type**
- What are the facts? **Revenue, Quantity, UnitPrice**

Datamart Design

Sales Datamart

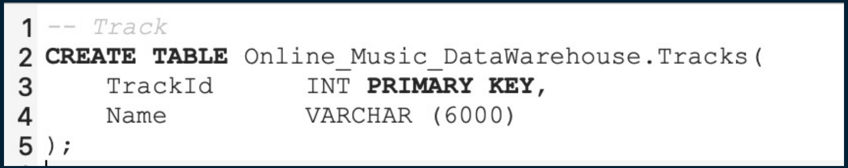
Draw the data model of your proposed DataMart (star schema).



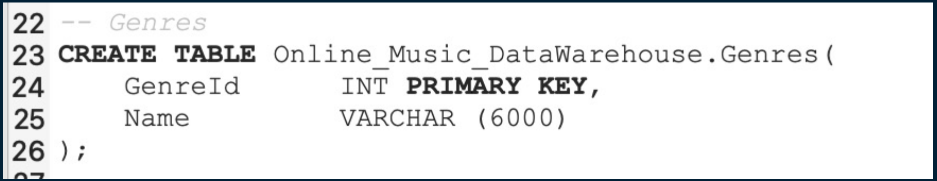
Data Warehouse Implementation

In a new SQLite database, define and create the tables for the new datamart. Paste screenshots of your DDL SQL statements on the following pages.

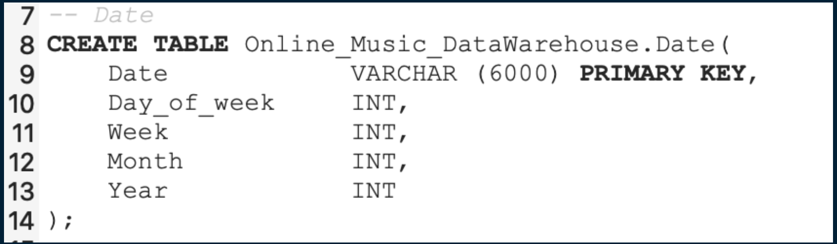
Dimension Tables Definition



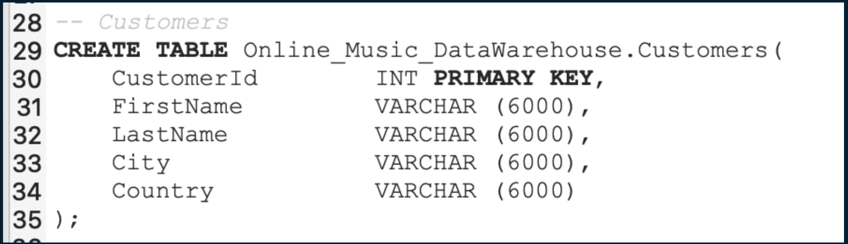
```sql
-- Track
CREATE TABLE Online_Music_DataWarehouse.Tracks(
    TrackId        INT PRIMARY KEY,
    Name           VARCHAR (6000)
);
```



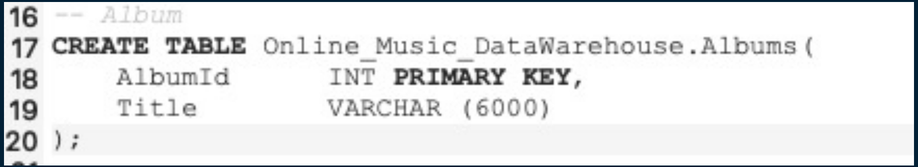
```sql
-- Genres
CREATE TABLE Online_Music_DataWarehouse.Genres(
    GenreId        INT PRIMARY KEY,
    Name           VARCHAR (6000)
);
```



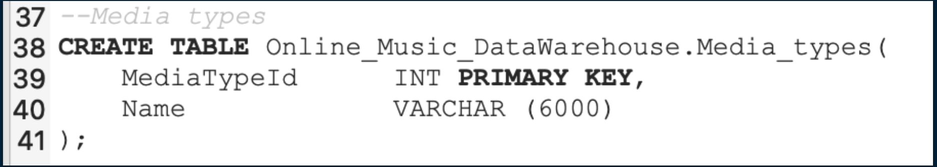
```sql
-- Date
CREATE TABLE Online_Music_DataWarehouse.Date(
    Date               VARCHAR (6000) PRIMARY KEY,
    Day_of_week        INT,
    Week               INT,
    Month              INT,
    Year               INT
);
```



```sql
-- Customers
CREATE TABLE Online_Music_DataWarehouse.Customers(
    CustomerId         INT PRIMARY KEY,
    FirstName          VARCHAR (6000),
    LastName           VARCHAR (6000),
    City               VARCHAR (6000),
    Country            VARCHAR (6000)
);
```

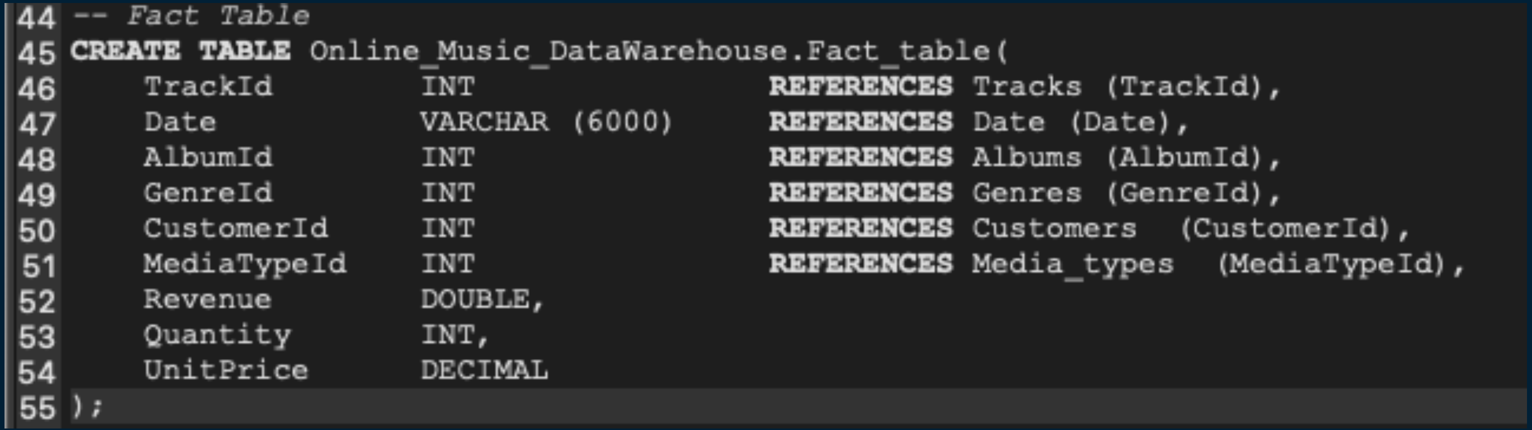


```sql
-- Album
CREATE TABLE Online_Music_DataWarehouse.Albums(
    AlbumId        INT PRIMARY KEY,
    Title          VARCHAR (6000)
);
```



```sql
--Media types
CREATE TABLE Online_Music_DataWarehouse.Media_types(
    MediaTypeId        INT PRIMARY KEY,
    Name               VARCHAR (6000)
);
```

Fact Tables Definition

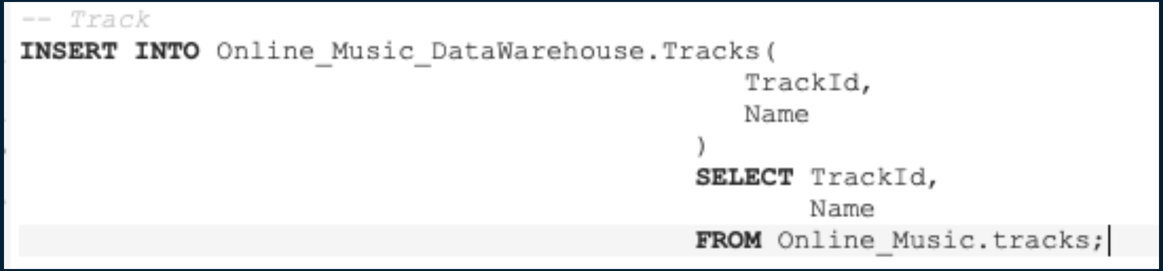


```
-- Fact Table
CREATE TABLE Online_Music_DataWarehouse.Fact_table(
    TrackId         INT                 REFERENCES Tracks (TrackId),
    Date            VARCHAR (6000)      REFERENCES Date (Date),
    AlbumId         INT                 REFERENCES Albums (AlbumId),
    GenreId         INT                 REFERENCES Genres (GenreId),
    CustomerId      INT                 REFERENCES Customers  (CustomerId),
    MediaTypeId     INT                 REFERENCES Media_types  (MediaTypeId),
    Revenue         DOUBLE,
    Quantity        INT,
    UnitPrice       DECIMAL
);
```

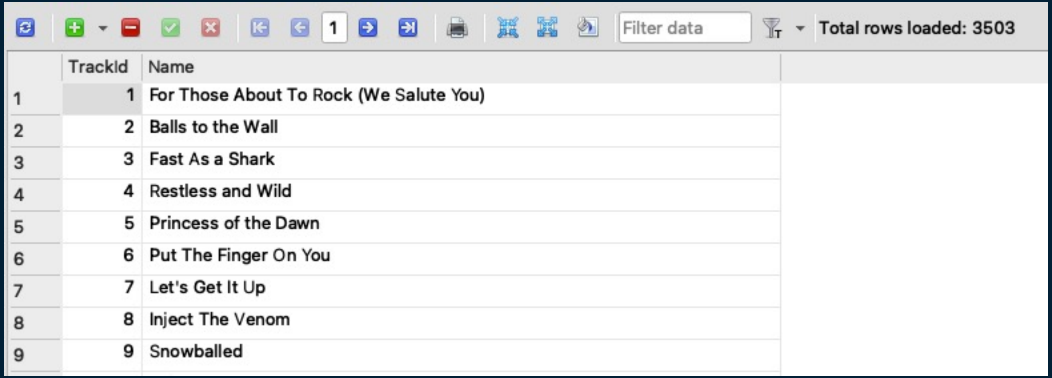
Data Warehouse Implementation

Write INSERT INTO SQL commands to populate the tables in your datamart. Paste screenshots of your SQL statements and resulting table contents onto the following pages

Tracks Table



```
-- Track
INSERT INTO Online_Music_DataWarehouse.Tracks(
                                            TrackId,
                                            Name
                                        )
                                        SELECT TrackId,
                                               Name
                                        FROM Online_Music.tracks;
```

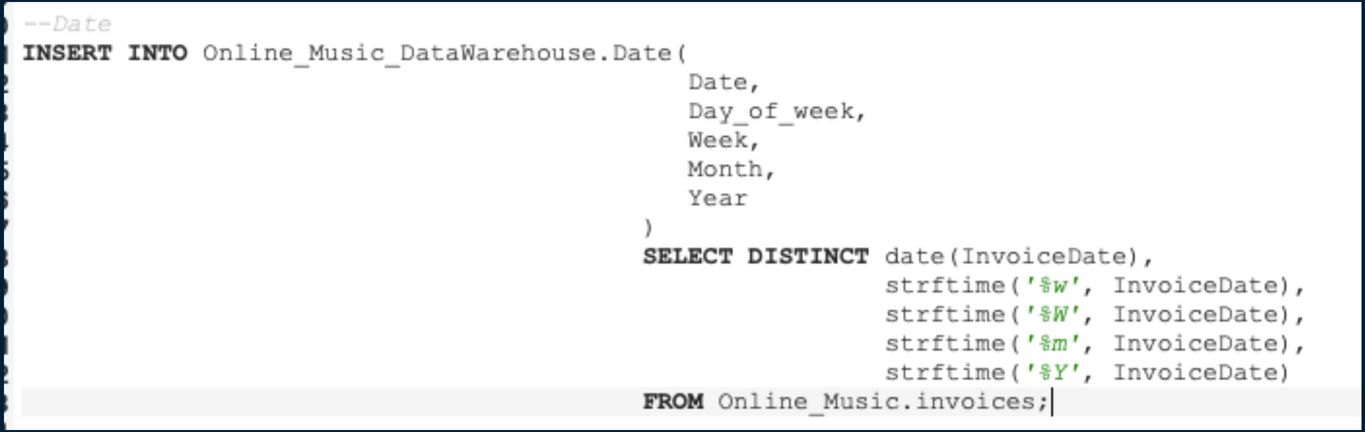


Filter data

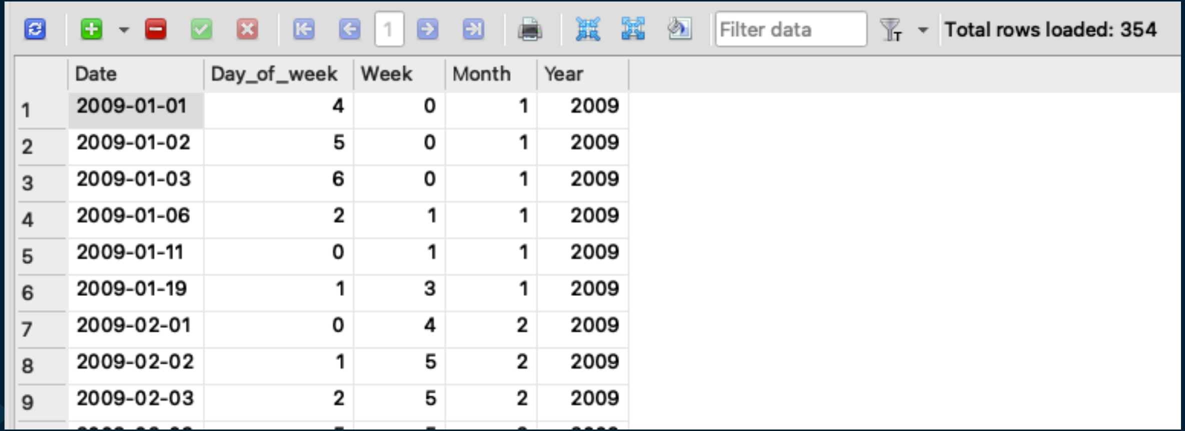
Total rows loaded: 3503

	TrackId	Name
1	1	For Those About To Rock (We Salute You)
2	2	Balls to the Wall
3	3	Fast As a Shark
4	4	Restless and Wild
5	5	Princess of the Dawn
6	6	Put The Finger On You
7	7	Let's Get It Up
8	8	Inject The Venom
9	9	Snowballed

Date Table



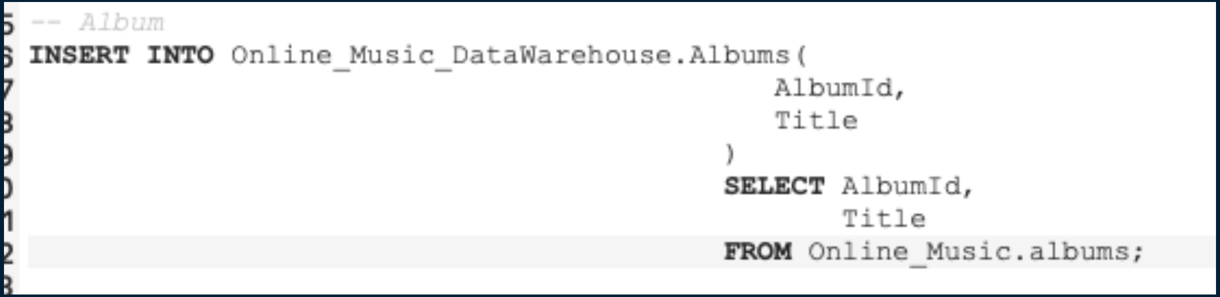
```
--Date
INSERT INTO Online_Music_DataWarehouse.Date(
                                            Date,
                                            Day_of_week,
                                            Week,
                                            Month,
                                            Year
                                         )
                                         SELECT DISTINCT date(InvoiceDate),
                                                         strftime('%w', InvoiceDate),
                                                         strftime('%W', InvoiceDate),
                                                         strftime('%m', InvoiceDate),
                                                         strftime('%Y', InvoiceDate)
                                         FROM Online_Music.invoices;
```



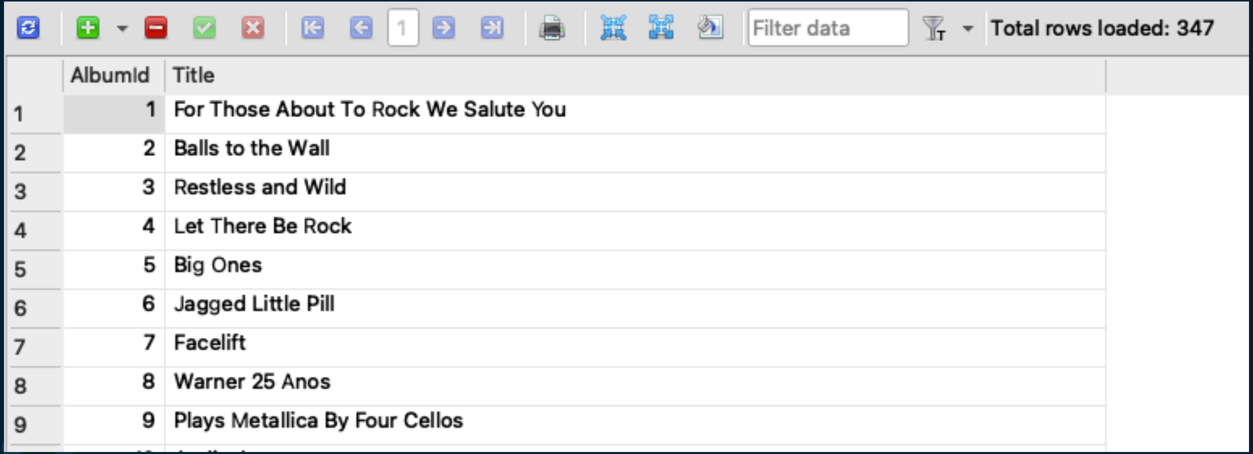
Total rows loaded: 354

	Date	Day_of_week	Week	Month	Year
1	2009-01-01	4	0	1	2009
2	2009-01-02	5	0	1	2009
3	2009-01-03	6	0	1	2009
4	2009-01-06	2	1	1	2009
5	2009-01-11	0	1	1	2009
6	2009-01-19	1	3	1	2009
7	2009-02-01	0	4	2	2009
8	2009-02-02	1	5	2	2009
9	2009-02-03	2	5	2	2009

Album Table



```
-- Album
INSERT INTO Online_Music_DataWarehouse.Albums(
                                                AlbumId,
                                                Title
                                             )
                                             SELECT AlbumId,
                                                    Title
                                             FROM Online_Music.albums;
```

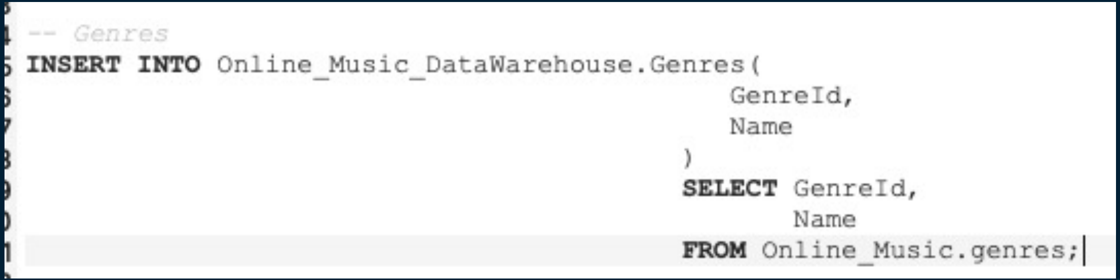


Filter data

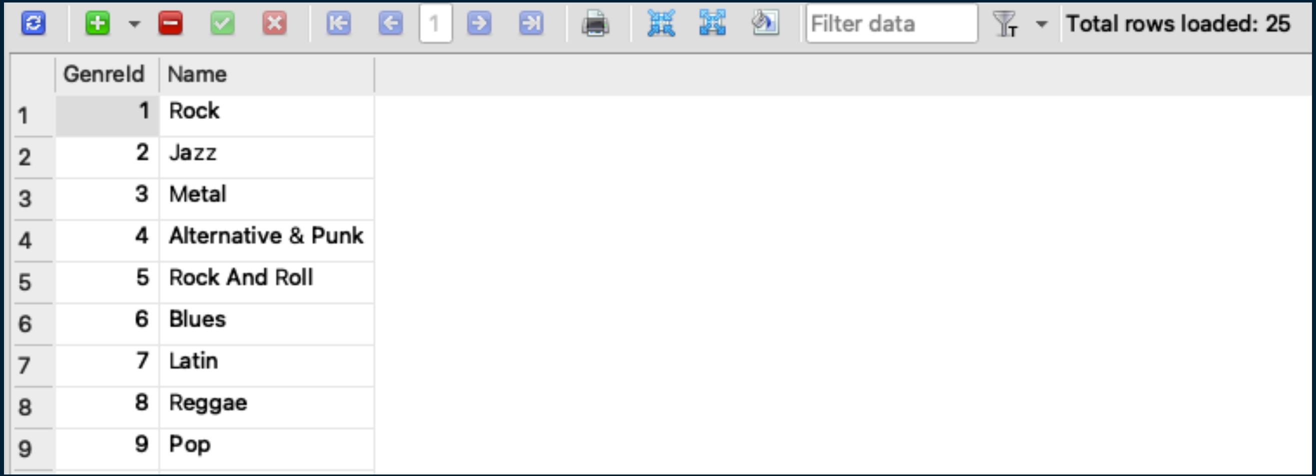
Total rows loaded: 347

	AlbumId	Title
1	1	For Those About To Rock We Salute You
2	2	Balls to the Wall
3	3	Restless and Wild
4	4	Let There Be Rock
5	5	Big Ones
6	6	Jagged Little Pill
7	7	Facelift
8	8	Warner 25 Anos
9	9	Plays Metallica By Four Cellos

Genres Table



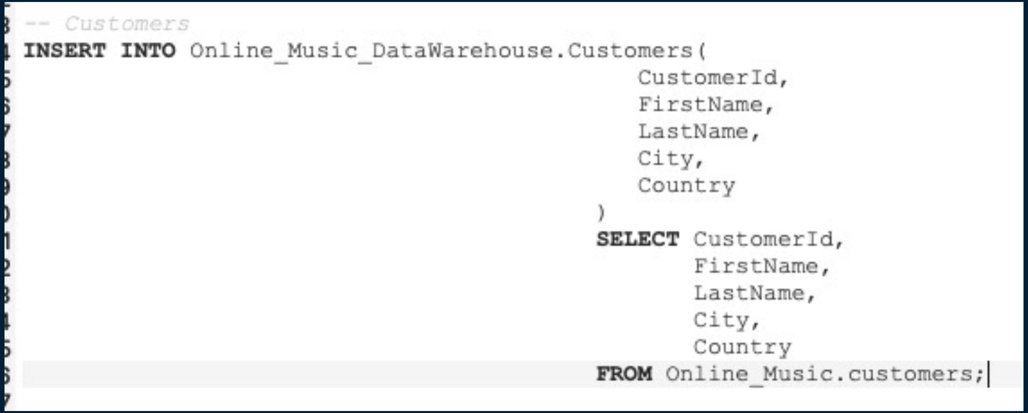
```
-- Genres
INSERT INTO Online_Music_DataWarehouse.Genres(
                                                     GenreId,
                                                     Name
                                                 )
                                                 SELECT GenreId,
                                                        Name
                                                 FROM Online_Music.genres;
```



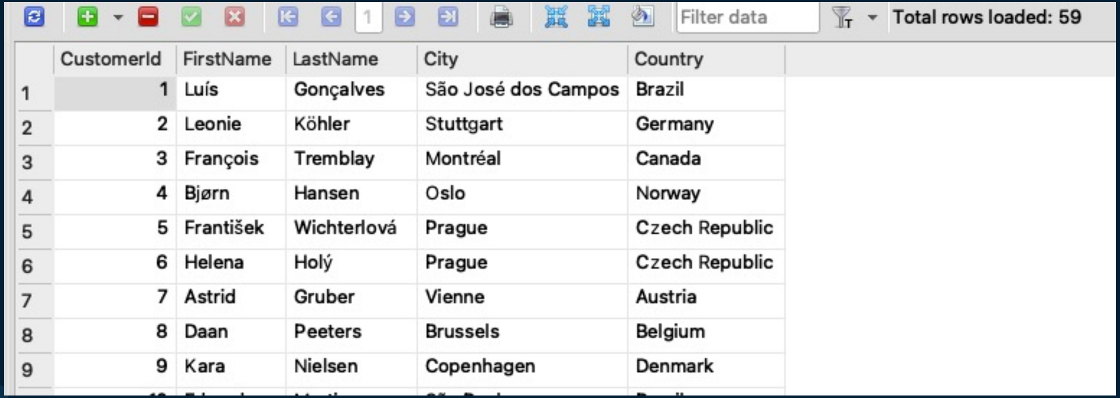
Total rows loaded: 25

	GenreId	Name
1	1	Rock
2	2	Jazz
3	3	Metal
4	4	Alternative & Punk
5	5	Rock And Roll
6	6	Blues
7	7	Latin
8	8	Reggae
9	9	Pop

Customers Table



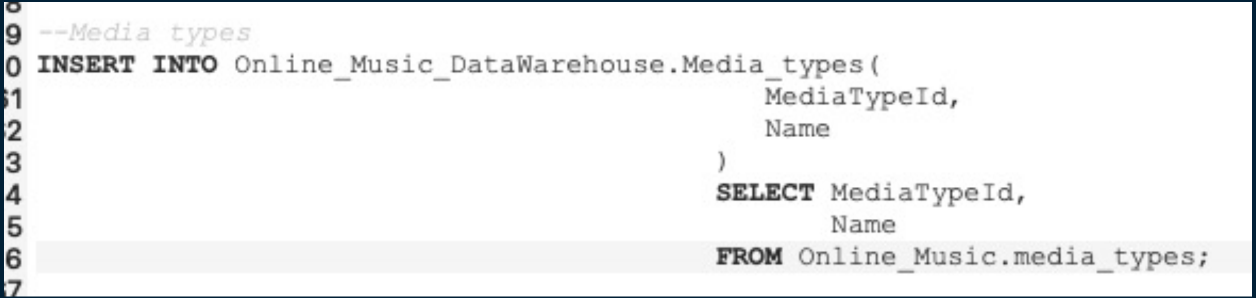
```
-- Customers
INSERT INTO Online_Music_DataWarehouse.Customers(
                                                        CustomerId,
                                                        FirstName,
                                                        LastName,
                                                        City,
                                                        Country
                                                    )
                                                    SELECT CustomerId,
                                                           FirstName,
                                                           LastName,
                                                           City,
                                                           Country
                                                    FROM Online_Music.customers;
```



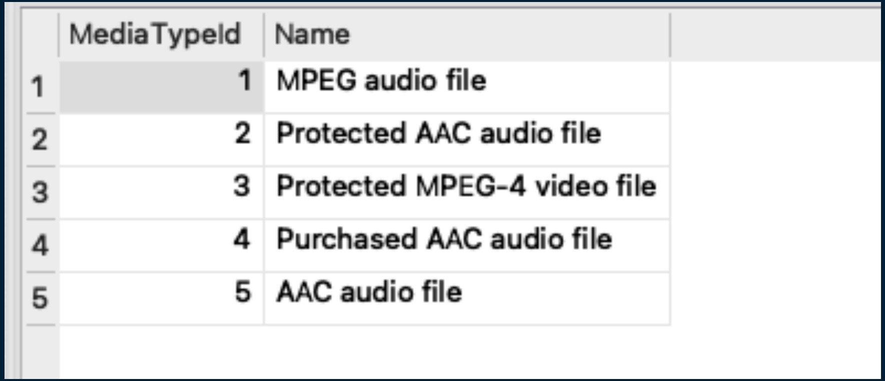
Filter data   Total rows loaded: 59

	CustomerId	FirstName	LastName	City	Country
1	1	Luís	Gonçalves	São José dos Campos	Brazil
2	2	Leonie	Köhler	Stuttgart	Germany
3	3	François	Tremblay	Montréal	Canada
4	4	Bjørn	Hansen	Oslo	Norway
5	5	František	Wichterlová	Prague	Czech Republic
6	6	Helena	Holý	Prague	Czech Republic
7	7	Astrid	Gruber	Vienne	Austria
8	8	Daan	Peeters	Brussels	Belgium
9	9	Kara	Nielsen	Copenhagen	Denmark

Media types Table

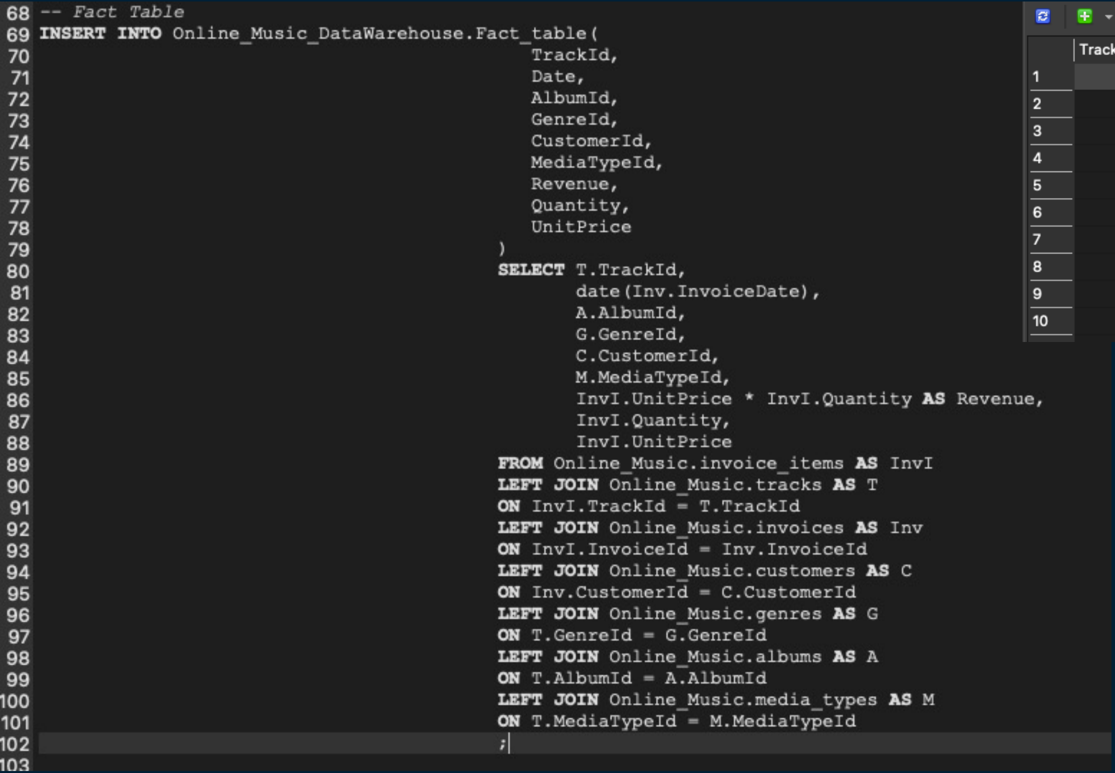


```
--Media types
INSERT INTO Online_Music_DataWarehouse.Media_types(
                                                                MediaTypeId,
                                                                Name
                                                            )
                                                            SELECT MediaTypeId,
                                                                   Name
                                                            FROM Online_Music.media_types;
```

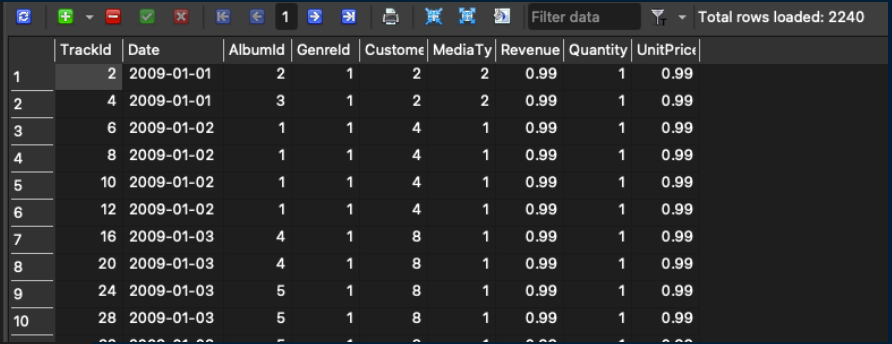


	MediaTypeId	Name
1	1	MPEG audio file
2	2	Protected AAC audio file
3	3	Protected MPEG-4 video file
4	4	Purchased AAC audio file
5	5	AAC audio file

Fact Table



```
-- Fact Table
INSERT INTO Online_Music_DataWarehouse.Fact_table(
                                                TrackId,
                                                Date,
                                                AlbumId,
                                                GenreId,
                                                CustomerId,
                                                MediaTypeId,
                                                Revenue,
                                                Quantity,
                                                UnitPrice
                                             )
                                             SELECT T.TrackId,
                                                    date(Inv.InvoiceDate),
                                                    A.AlbumId,
                                                    G.GenreId,
                                                    C.CustomerId,
                                                    M.MediaTypeId,
                                                    InvI.UnitPrice * InvI.Quantity AS Revenue,
                                                    InvI.Quantity,
                                                    InvI.UnitPrice
                                             FROM Online_Music.invoice_items AS InvI
                                             LEFT JOIN Online_Music.tracks AS T
                                             ON InvI.TrackId = T.TrackId
                                             LEFT JOIN Online_Music.invoices AS Inv
                                             ON InvI.InvoiceId = Inv.InvoiceId
                                             LEFT JOIN Online_Music.customers AS C
                                             ON Inv.CustomerId = C.CustomerId
                                             LEFT JOIN Online_Music.genres AS G
                                             ON T.GenreId = G.GenreId
                                             LEFT JOIN Online_Music.albums AS A
                                             ON T.AlbumId = A.AlbumId
                                             LEFT JOIN Online_Music.media_types AS M
                                             ON T.MediaTypeId = M.MediaTypeId
                                             ;
```

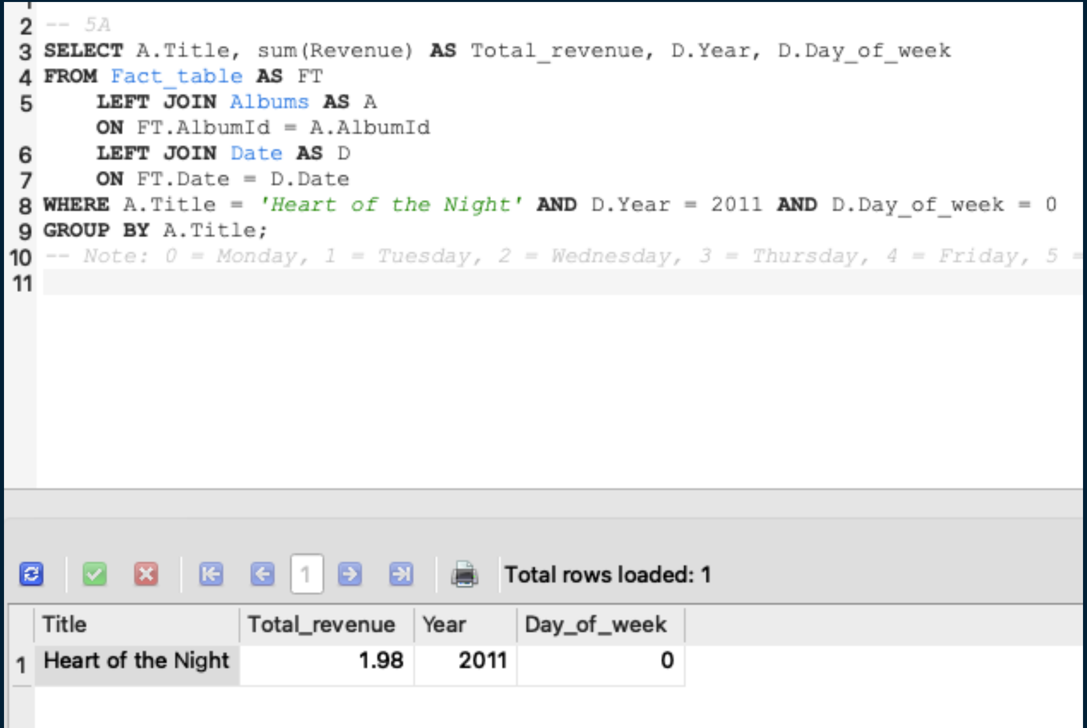


1 | Filter data | Total rows loaded: 2240

	TrackId	Date	AlbumId	GenreId	Custome	MediaTy	Revenue	Quantity	UnitPric
1	2	2009-01-01	2	1	2	2	0.99	1	0.99
2	4	2009-01-01	3	1	2	2	0.99	1	0.99
3	6	2009-01-02	1	1	4	1	0.99	1	0.99
4	8	2009-01-02	1	1	4	1	0.99	1	0.99
5	10	2009-01-02	1	1	4	1	0.99	1	0.99
6	12	2009-01-02	1	1	4	1	0.99	1	0.99
7	16	2009-01-03	4	1	8	1	0.99	1	0.99
8	20	2009-01-03	4	1	8	1	0.99	1	0.99
9	24	2009-01-03	5	1	8	1	0.99	1	0.99
10	28	2009-01-03	5	1	8	1	0.99	1	0.99

Data Warehouse Queries

What was the total revenue for the album “Heart of the Night” on Mondays in the year 2011?



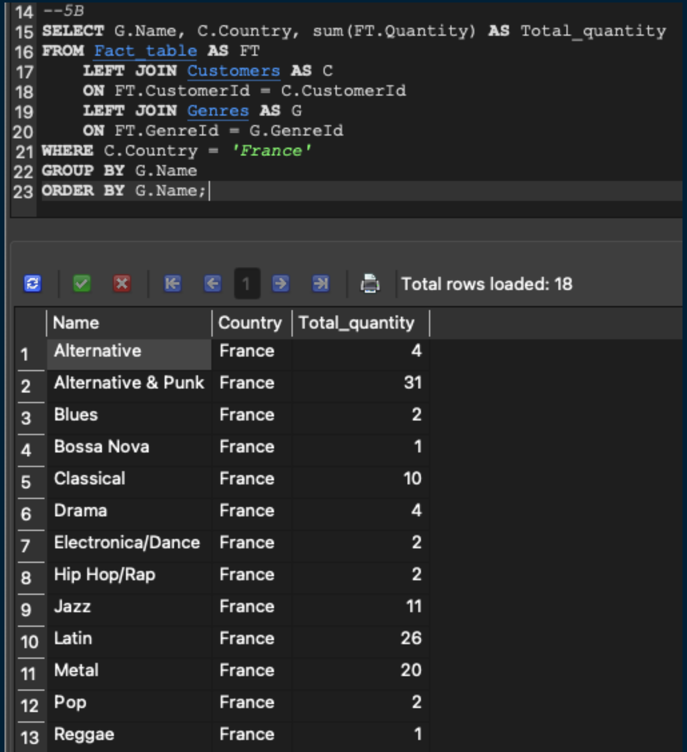
```
-- 5A
SELECT A.Title, sum(Revenue) AS Total_revenue, D.Year, D.Day_of_week
FROM Fact_table AS FT
     LEFT JOIN Albums AS A
     ON FT.AlbumId = A.AlbumId
     LEFT JOIN Date AS D
     ON FT.Date = D.Date
WHERE A.Title = 'Heart of the Night' AND D.Year = 2011 AND D.Day_of_week = 0
GROUP BY A.Title;
-- Note: 0 = Monday, 1 = Tuesday, 2 = Wednesday, 3 = Thursday, 4 = Friday, 5 =
```

Total rows loaded: 1

	Title	Total_revenue	Year	Day_of_week
1	Heart of the Night	1.98	2011	0

Data Warehouse Queries

Create a report showing the total quantity of tracks sold in France for each category sorted by category name.



```
--5B
SELECT G.Name, C.Country, sum(FT.Quantity) AS Total_quantity
FROM Fact_table AS FT
    LEFT JOIN Customers AS C
    ON FT.CustomerId = C.CustomerId
    LEFT JOIN Genres AS G
    ON FT.GenreId = G.GenreId
WHERE C.Country = 'France'
GROUP BY G.Name
ORDER BY G.Name;
```

Total rows loaded: 18

	Name	Country	Total_quantity
1	Alternative	France	4
2	Alternative & Punk	France	31
3	Blues	France	2
4	Bossa Nova	France	1
5	Classical	France	10
6	Drama	France	4
7	Electronica/Dance	France	2
8	Hip Hop/Rap	France	2
9	Jazz	France	11
10	Latin	France	26
11	Metal	France	20
12	Pop	France	2
13	Reggae	France	1

Aggregate Fact Tables

- Over time, your datamart has gotten very large and query performance is starting to degrade. You are considering making additional fact tables that are aggregated alone one or more dimensions.
- On the following pages:
  - Identify an aggregate table for each of your dimensions
  - Identify an aggregate table for each possible pair of your dimensions

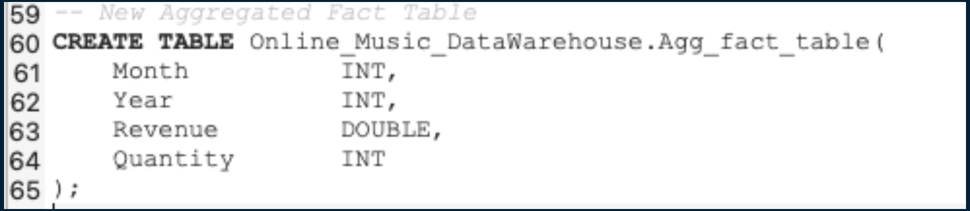
Aggregate Fact Tables – Single Attribute Aggregations

- Track’s name by date by album by genre by customer by media type
- Day of week by track’s name by album by genre by customer by media type
- Month by track’s name by album by genre by customer by media type
- Year by track’s name by album by genre by customer by media type
- Album’s title by track’s name by date by genre by customer by media type
- Genre’s name by track’s name by date by album by customer by media type
- Customer’s city by track’s name by date by album by genre by media type
- Customer’s country by track’s name by date by album by genre by media type
- Media type’s name by by track’s name by date by album by genre by customer

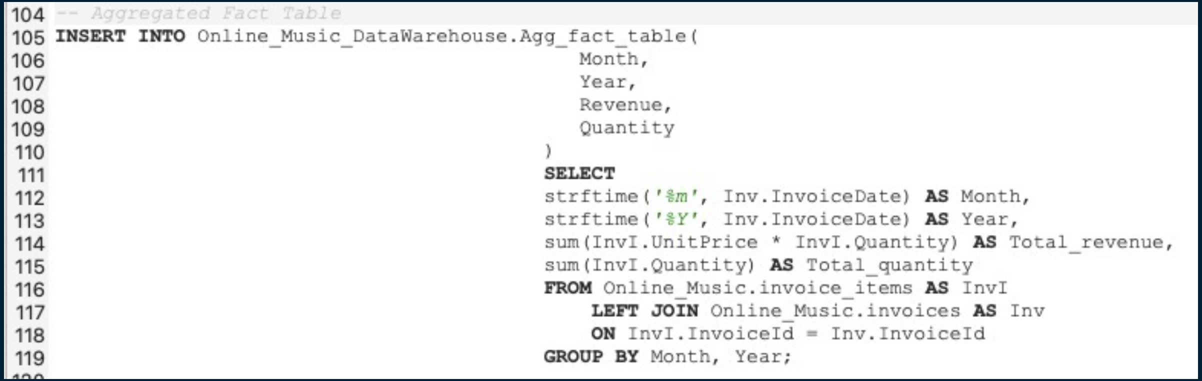
Aggregate Fact Tables – Dual Attribute Aggregations

- Track's name and week by album by genre by customer by media type
- Album's title and month by track's name by genre by customer by media type
- Genre's name and customer's country by track's name by date by album by media type
- Genre's name and year by track's name by album by customer by media type

Using SQLite, create and populate a new aggregated fact table that aggregates the date to be at the month level.

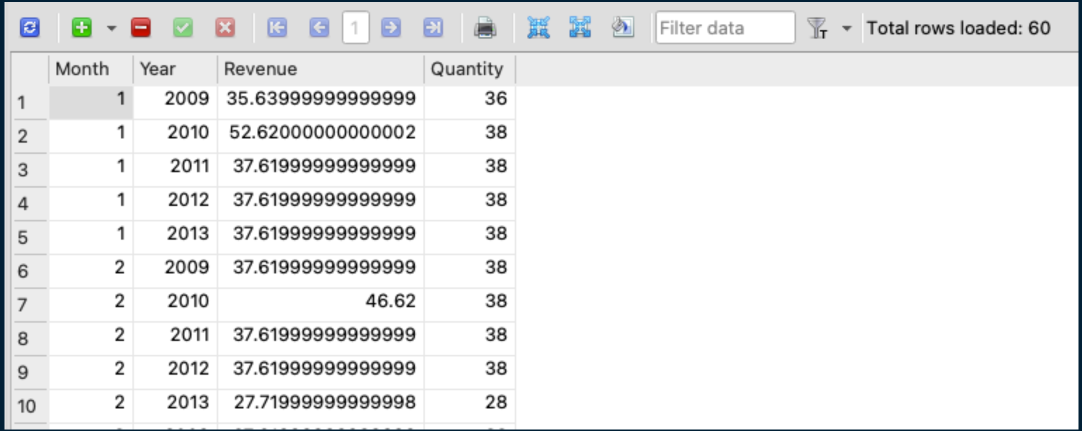


```sql
-- New Aggregated Fact Table
CREATE TABLE Online_Music_DataWarehouse.Agg_fact_table(
    Month           INT,
    Year            INT,
    Revenue         DOUBLE,
    Quantity        INT
);
```



```sql
-- Aggregated Fact Table
INSERT INTO Online_Music_DataWarehouse.Agg_fact_table(
                                                                Month,
                                                                Year,
                                                                Revenue,
                                                                Quantity
                                                            )
                                                            SELECT
                                                            strftime('%m', Inv.InvoiceDate) AS Month,
                                                            strftime('%Y', Inv.InvoiceDate) AS Year,
                                                            sum(InvI.UnitPrice * InvI.Quantity) AS Total_revenue,
                                                            sum(InvI.Quantity) AS Total_quantity
                                                            FROM Online_Music.invoice_items AS InvI
                                                                LEFT JOIN Online_Music.invoices AS Inv
                                                                ON InvI.InvoiceId = Inv.InvoiceId
                                                            GROUP BY Month, Year;
```

INSERT result



Filter data

Total rows loaded: 60

	Month	Year	Revenue	Quantity
1	1	2009	35.63999999999999	36
2	1	2010	52.62000000000002	38
3	1	2011	37.61999999999999	38
4	1	2012	37.61999999999999	38
5	1	2013	37.61999999999999	38
6	2	2009	37.61999999999999	38
7	2	2010	46.62	38
8	2	2011	37.61999999999999	38
9	2	2012	37.61999999999999	38
10	2	2013	27.71999999999998	28

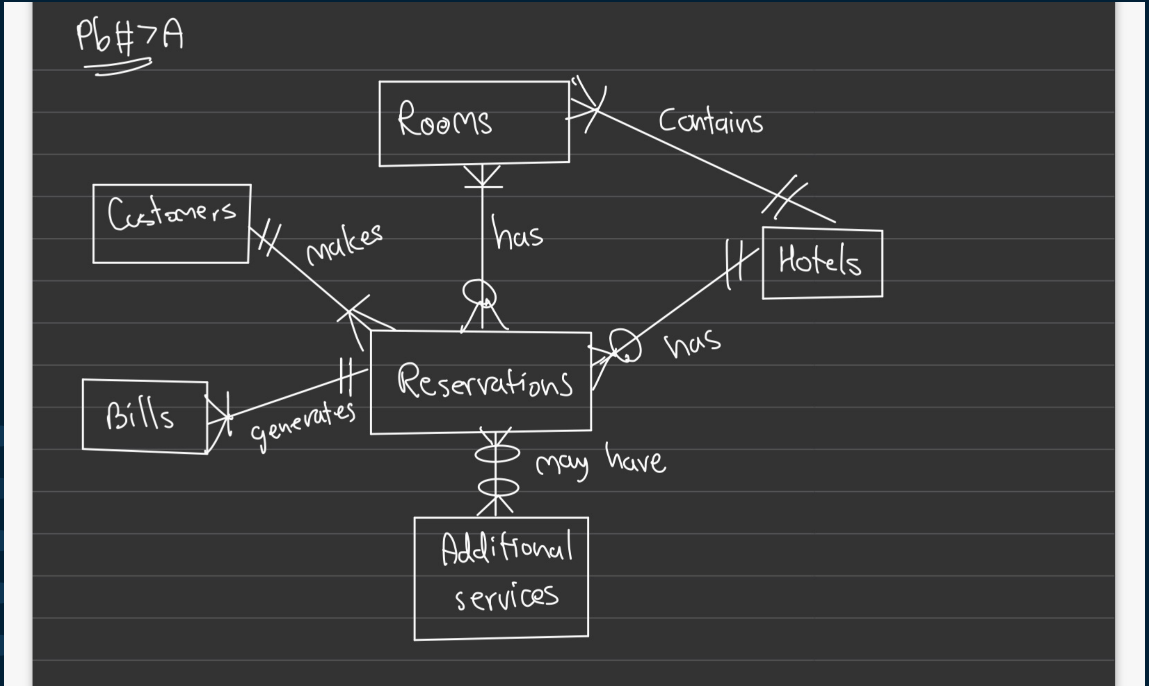
Conceptual Data Modeling

For this practice, I will develop a conceptual data model (entities and relationships only/no attributes) for a hotel reservation system for a large hotel chain. Here is a basic description of the business:

- There are multiple hotels in the system
- Customers can make reservations for rooms in specific hotels. A reservation may be for more than one room. Every reservation belongs to one customer.
- During their stay at the hotel, customers can purchase additional services (food, spa, etc.) which get added to the reservation
- When the customer checks out, a final bill is prepared based on the reservation records.

Conceptual Data Modeling – ER Diagram

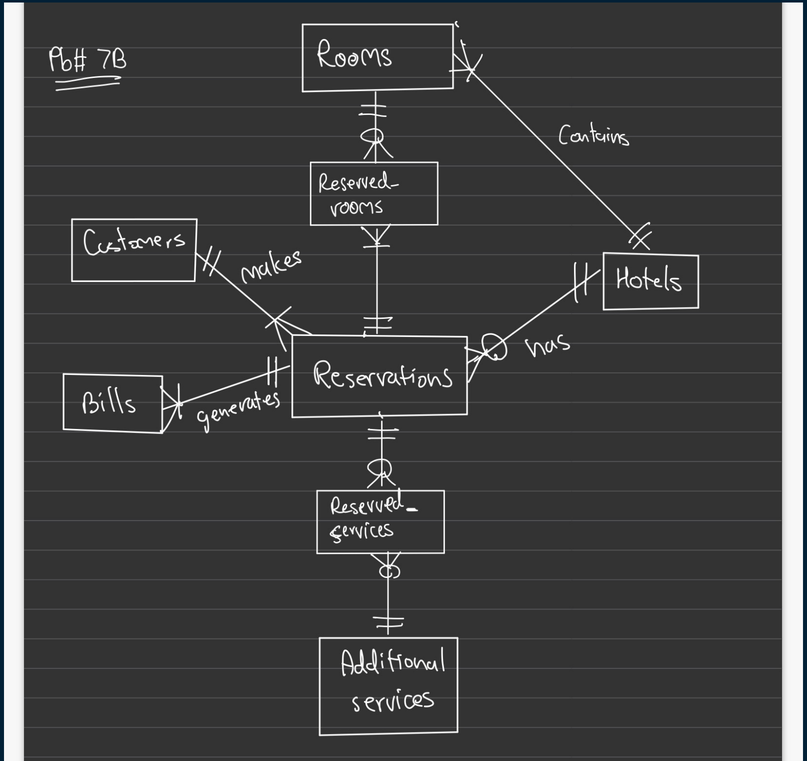
Using the crow's feet notation from class, draw the ER Diagram for your conceptual data model (attributes not required) below.



- I define a reservation can generate 1 or many bills because there might be something wrong during check out. Therefore, it is better to prepare for the situation.
- I do not connect customers to rooms and hotels since we can get the information in reservations.
- Accoding to hotel booking websites, I can only reserve for 1 hotel per booking. Thus, I define a reservation can have 1 and only 1 hotel.

Conceptual Data Modeling – ER Diagram

Now, re-draw your ER diagram to resolve any many-to-many relationships



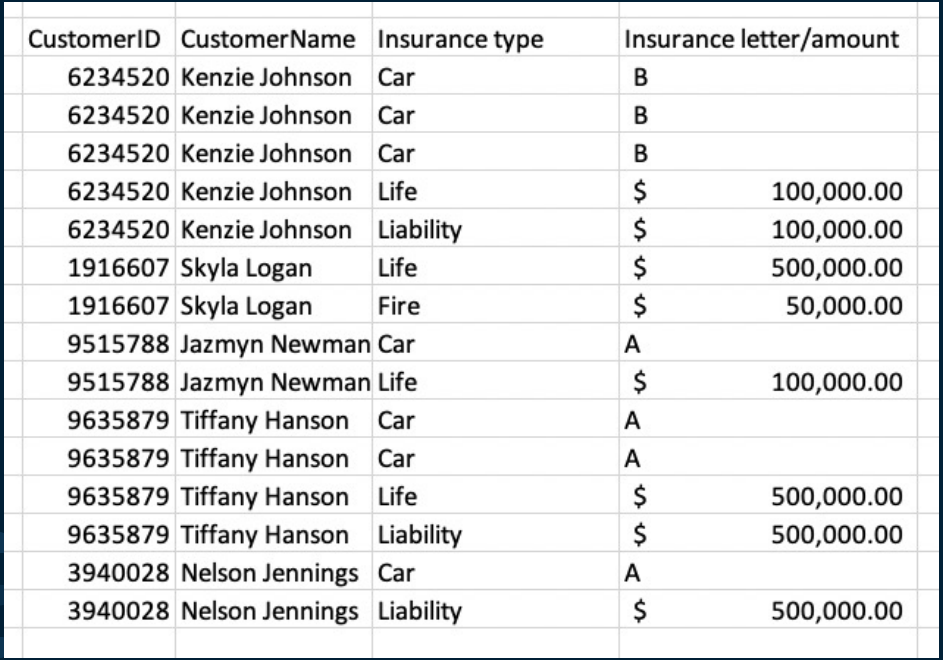
Dataset Structuring

The dataset below is from an insurance company and contains information on policies held by customers. The car insurance policy contains a letter indicating the type of policy the customer holds. For the life, fire, and liability insurance, the table contains the amount of the policy

On the following page, convert this dataset to a multiple-rows-per-subject dataset structure

Customer ID	Customer Name	Car #1	Car #2	Car #3	Life	Fire	Liability
6234520	Kenzie Johnson	B	B	B	$ 100,000		$ 100,000
1916607	Skyla Logan				$ 500,000	$ 50,000	
9515788	Jazmyn Newman	A			$ 100,000		
9635879	Tiffany Hanson	A	A		$ 500,000		$ 500,000
3940028	Nelson Jennings	A					$ 500,000

Dataset Structuring



CustomerID	CustomerName	Insurance type	Insurance letter/amount
6234520	Kenzie Johnson	Car	B
6234520	Kenzie Johnson	Car	B
6234520	Kenzie Johnson	Car	B
6234520	Kenzie Johnson	Life	$ 100,000.00
6234520	Kenzie Johnson	Liability	$ 100,000.00
1916607	Skyla Logan	Life	$ 500,000.00
1916607	Skyla Logan	Fire	$ 50,000.00
9515788	Jazmyn Newman	Car	A
9515788	Jazmyn Newman	Life	$ 100,000.00
9635879	Tiffany Hanson	Car	A
9635879	Tiffany Hanson	Car	A
9635879	Tiffany Hanson	Life	$ 500,000.00
9635879	Tiffany Hanson	Liability	$ 500,000.00
3940028	Nelson Jennings	Car	A
3940028	Nelson Jennings	Liability	$ 500,000.00

- Although, it seems weird to combine quantitative and qualitative data into the same column, it is the tidiest way to combine the data into a multiple-rows-per-subject dataset.