CODE FOR SQL PROJECT

TITLE: Music Store Relational Database

```
-- Artist Table
CREATE TABLE artist (
    artist_id INTEGER PRIMARY KEY,
    name TEXT
);

-- Album Table
CREATE TABLE album (
    album_id INTEGER PRIMARY KEY,
    title TEXT,
    artist_id INTEGER REFERENCES artist(artist_id)
);

-- Genre Table
CREATE TABLE genre (
    genre_id INTEGER PRIMARY KEY,
    name TEXT
);

-- MediaType Table
CREATE TABLE media_type (
    media_type_id INTEGER PRIMARY KEY,
    name TEXT
);

-- Track Table
CREATE TABLE track (
```

```
    track_id INTEGER PRIMARY KEY,
    name TEXT,
    album_id INTEGER REFERENCES album(album_id),
    media_type_id INTEGER REFERENCES media_type(media_type_id),
    genre_id INTEGER REFERENCES genre(genre_id),
    composer TEXT,
    milliseconds INTEGER,
    bytes INTEGER,
    unit_price NUMERIC
);

-- Playlist Table
CREATE TABLE playlist (
    playlist_id INTEGER PRIMARY KEY,
    name TEXT
);

-- PlaylistTrack Table
CREATE TABLE playlist_track (
    playlist_id INTEGER REFERENCES playlist(playlist_id),
    track_id INTEGER REFERENCES track(track_id),
    PRIMARY KEY (playlist_id, track_id)
);

-- Employee Table
CREATE TABLE employee (
    employee_id INTEGER PRIMARY KEY,
    last_name TEXT,
    first_name TEXT,
    title TEXT,
    reports_to INTEGER REFERENCES employee(employee_id),
```

```
    birth_date DATE,
    hire_date DATE,
    address TEXT,
    city TEXT,
    state TEXT,
    country TEXT,
    postal_code TEXT,
    phone TEXT,
    fax TEXT,
    email TEXT
);

-- Customer Table
CREATE TABLE customer (
    customer_id INTEGER PRIMARY KEY,
    first_name TEXT,
    last_name TEXT,
    company TEXT,
    address TEXT,
    city TEXT,
    state TEXT,
    country TEXT,
    postal_code TEXT,
    phone TEXT,
    fax TEXT,
    email TEXT,
    support_rep_id INTEGER REFERENCES employee(employee_id)
);

-- Invoice Table
CREATE TABLE invoice (
```

```
    invoice_id INTEGER PRIMARY KEY,
    customer_id INTEGER REFERENCES customer(customer_id),
    invoice_date DATE,
    billing_address TEXT,
    billing_city TEXT,
    billing_state TEXT,
    billing_country TEXT,
    billing_postal_code TEXT,
    total NUMERIC
);

-- InvoiceLine Table
CREATE TABLE invoice_line (
    invoice_line_id INTEGER PRIMARY KEY,
    invoice_id INTEGER REFERENCES invoice(invoice_id),
    track_id INTEGER REFERENCES track(track_id),
    unit_price NUMERIC,
    quantity INTEGER
);
\COPY artist FROM 'artist.csv' DELIMITER ',' CSV HEADER;
\COPY album FROM 'album2.csv' DELIMITER ',' CSV HEADER;
\COPY genre FROM 'genre.csv' DELIMITER ',' CSV HEADER;
\COPY media_type FROM 'media_type.csv' DELIMITER ',' CSV HEADER;
\COPY track FROM 'track.csv' DELIMITER ',' CSV HEADER;
\COPY playlist FROM 'playlist.csv' DELIMITER ',' CSV HEADER;
\COPY playlist_track FROM 'playlist_track.csv' DELIMITER ',' CSV HEADER;
\COPY employee FROM 'employee.csv' DELIMITER ',' CSV HEADER;
\COPY customer FROM 'customer.csv' DELIMITER ',' CSV HEADER;
\COPY invoice FROM 'invoice.csv' DELIMITER ',' CSV HEADER;
\COPY invoice_line FROM 'invoice_line.csv' DELIMITER ',' CSV HEADER;
```

CODE FOR ANSWERS:

EASY LEVEL:

```
-- Q1: Most senior employee
SELECT * FROM employee ORDER BY title DESC LIMIT 1;


-- Q2: Countries with the most invoices
SELECT billing_country, COUNT(*) AS invoice_count
FROM invoice
GROUP BY billing_country
ORDER BY invoice_count DESC;


-- Q3: Top 3 invoice totals
SELECT invoice_id, total
FROM invoice
ORDER BY total DESC
LIMIT 3;


-- Q4: City with highest total invoice amount
SELECT billing_city, SUM(total) AS total_amount
FROM invoice
GROUP BY billing_city
ORDER BY total_amount DESC
LIMIT 1;


-- Q5: Customer who spent the most
SELECT c.customer_id, c.first_name, c.last_name, SUM(i.total) AS total_spent
FROM customer c
JOIN invoice i ON c.customer_id = i.customer_id
GROUP BY c.customer_id, c.first_name, c.last_name
```

```
ORDER BY total_spent DESC
LIMIT 1;
```

MODERATE LEVEL:

```
-- Q1: Customers who listen to Rock
SELECT DISTINCT c.customer_id, c.first_name, c.last_name
FROM customer c
JOIN invoice i ON c.customer_id = i.customer_id
JOIN invoice_line il ON i.invoice_id = il.invoice_id
JOIN track t ON il.track_id = t.track_id
JOIN genre g ON t.genre_id = g.genre_id
WHERE g.name = 'Rock';

-- Q2: Top 10 Rock artists by track count
SELECT a.name AS artist_name, COUNT(t.track_id) AS rock_track_count
FROM artist a
JOIN album al ON a.artist_id = al.artist_id
JOIN track t ON al.album_id = t.album_id
JOIN genre g ON t.genre_id = g.genre_id
WHERE g.name = 'Rock'
GROUP BY a.name
ORDER BY rock_track_count DESC
LIMIT 10;

-- Q3: Tracks longer than average length
SELECT name, milliseconds
FROM track
WHERE milliseconds > (
    SELECT AVG(milliseconds) FROM track
)
ORDER BY milliseconds DESC;
```

HARD LEVEL:

-- Q1: Customer spending per artist

```
\COPY (SELECT c.customer_id, c.first_name, c.last_name, a.artist_id, a.name AS artist_name, SUM(il.unit_price * il.quantity) AS total_spent FROM customer c JOIN invoice i ON c.customer_id = i.customer_id JOIN invoice_line il ON i.invoice_id = il.invoice_id JOIN track t ON il.track_id = t.track_id JOIN album al ON t.album_id = al.album_id JOIN artist a ON al.artist_id = a.artist_id GROUP BY c.customer_id, c.first_name, c.last_name, a.artist_id, a.name ORDER BY c.customer_id, total_spent DESC) TO 'C:/Users/Ansh/Desktop/customer_artist_sales.csv' WITH CSV HEADER;
```

-- Q2: Most popular genre per country

```
WITH genre_sales AS (
    SELECT
        c.country,
        g.name AS genre_name,
        COUNT(*) AS purchase_count,
        RANK() OVER (PARTITION BY c.country ORDER BY COUNT(*) DESC) AS genre_rank
    FROM customer c
    JOIN invoice i ON c.customer_id = i.customer_id
    JOIN invoice_line il ON i.invoice_id = il.invoice_id
    JOIN track t ON il.track_id = t.track_id
    JOIN genre g ON t.genre_id = g.genre_id
    GROUP BY c.country, g.name
)
SELECT country, genre_name, purchase_count
FROM genre_sales
WHERE genre_rank = 1;
```

-- Q3: Top-spending customer per country

```
WITH customer_spending AS (
    SELECT
        c.country,
        c.customer_id,
        c.first_name,
```

```
    c.last_name,
    SUM(i.total) AS total_spent,
    RANK() OVER (PARTITION BY c.country ORDER BY SUM(i.total) DESC) AS rank
  FROM customer c
  JOIN invoice i ON c.customer_id = i.customer_id
  GROUP BY c.country, c.customer_id, c.first_name, c.last_name
)
SELECT * FROM customer_spending
WHERE rank = 1;
```