ITMD 526 14TH MARCH 2025

DATAWAREHOUSE IMPLEMENTATION PROJECT

ILLINOIS INSTITUTE OF TECHNOLOGY

BHUVAN MYSORE SRIDHAR

A20572490 **Guidance:** Prof. Raj Krishnan

Introduction

The main goal of this project involved designing a Music Playlist Database for Snowflake which connected to Azure Data Lake Storage (ADLS). The project sought to establish a high-performance data warehouse system which efficiently stores and handles and analyzes music-related data on a scalable level.

The integration of Snowflake cloud platform with ADLS storage delivered unobstructed data processing and integration capabilities for our system. The data handling process included three levels of Bronze Silver Gold architecture that handled data entry and data manipulation and data analysis respectively.

This document covers the entire process of building the database infrastructure and describing Azure-Snowflake integration alongside significant findings extracted from processed data. The document explains implementing methods as well as access control systems for roles and advanced querying approaches at their core.

DATA SOURCE:

FILE NAME	DESCRIPTION
Album.csv	Contains data related to albums, including columns such as `AlbumId`, `Title`, `ArtistId`.
Artist.csv	Contains data related to artists, including columns such as `ArtistId`, `Name`.
Customer.csv	Contains data related to customers, including columns such as `CustomerId`, `FirstName`, `LastName`, `Company`, `Address`.
Employee.csv	Contains data related to employees, including columns such as `EmployeeId`, `LastName`, `FirstName`, `Title`, `ReportsTo`.
Genre.csv	Contains data related to music genres, including columns such as `GenreId`, `Name`.

Invoice.csv	Contains data related to invoices, including columns such as `InvoiceId`, `CustomerId`, `InvoiceDate`, `BillingAddress`, `BillingCity`.
InvoiceLine.csv	Contains data related to invoice details, including columns such as `InvoiceLineId`, `InvoiceId`, `TrackId`, `UnitPrice`, `Quantity`.
MediaType.csv	Contains data related to media types, including columns such as `MediaTypeId`, `Name`.
Playlist.csv	Contains data related to playlists, including columns such as `PlaylistId`, `Name`.
PlaylistTrack.csv	Contains data mapping tracks to playlists, including columns such as `PlaylistId`, `TrackId`.
Track.csv	Contains data related to tracks, including columns such as `TrackId`, `Name`, `AlbumId`, `MediaTypeId`, `GenreId`.

---

1. Database Architecture

Overview of the Three-Tier Data Architecture

The standard data processing system includes Bronze and Silver and Gold layers which follow a predefined blueprint for data ingestion and transformation and reporting tasks. This architecture provides:

- **Separation of concerns:** The data processing system consists of various layers with unique functions which maintain clean organized data and optimal performance.
- **Performance optimization:** The system's stacked layers conduct limited operations for maximizing the speed of data retrieval requests.
- **Scalability:** Layers allow us to keep control over large data collections in an efficient manner.

1.1 Bronze Layer (Raw Data Storage)

The primary function of the Bronze Layer consists of raw data storage via Azure Data Lake Storage (ADLS) to receive data from various sources in its initial format. The raw data arrives at this layer which serves as a storage location before the data operations commence.

Characteristics of the Bronze Layer:

- Stores historical records in their raw format.

- It preserves duplicate and missing data before handling them in subsequent stages of data processing.
- The storage system adds data through append-only operations which prevents modification of previously recorded information.
- The system enables the storage of semi-structured and structured data types obtained from various data sources.

Example Table in the Bronze Layer:

```
CREATE TABLE BRONZE.RAW_TRACK (

    TrackId INT PRIMARY KEY,

    Name STRING,

    AlbumId INT,

    MediaTypeId INT,

    GenreId INT,

    Composer STRING,

    Milliseconds INT,

    Bytes INT,

    UnitPrice FLOAT,

    ingestion_timestamp TIMESTAMP_NTZ DEFAULT CURRENT_TIMESTAMP()

);
```

The track data collected from the source system resides within this table before validation or cleansing procedures begin.

1.2 Silver Layer (Cleansed and Transformed Data)

The Silver Layer organizes raw data through three processes of data validation followed by deduplication and data structuring. The refinement process in the Silver Layer creates consistent and clean data which satisfies analytical requirements.

Characteristics of the Silver Layer:

- Cleans inconsistent or missing values.
- Data processing through this method delivers standardized information ready for business utilization.

- The program establishes important key relationships between tables.
- The system follows incremental processing it uses for tracking historical changes.

Example Transformation from Bronze to Silver Layer:

```
INSERT INTO SILVER.DIM_TRACK (TrackId, AlbumId, Name, Composer,
Media_Type_Name, Genre_Name, Milliseconds, Bytes, UnitPrice)

SELECT DISTINCT

  Tk.TrackId, Tk.AlbumId, Tk.Name, Tk.Composer,

  Media.Name AS Media_Type_Name,

  Gen.Name AS Genre_Name,

  Tk.Milliseconds, Tk.Bytes, Tk.UnitPrice

FROM BRONZE.RAW_TRACK Tk

INNER JOIN BRONZE.RAW_MEDIATYPE Media ON Tk.MediaTypeId =
Media.MediaType_Id

INNER JOIN BRONZE.RAW_GENRE Gen ON Tk.GenreId = Gen.GenreId;
```

This transformation:

- **Joins multiple tables** to enrich data.
- **Removes duplicate records** for accuracy.
- **Adds meaningful attributes** such as `Media_Type_Name` and `Genre_Name`.

1.3 Gold Layer (Aggregated and Analytical Data)

Advanced analytics together with reporting capabilities as well as business intelligence functions are optimized within the Gold Layer. The Gold Layer includes standardized data that functions for efficient query processing.

Characteristics of the Gold Layer:

- The system stores three metrics together: revenue per track, total sales and customer lifetime value.
- The system includes optimized structures which enhance efficient data querying abilities.
- The database system enables both analysis of data trends and calculation of performance indicators.
- The system depends on fact and dimension tables to enable structured reporting.

Example Gold Layer Table:

```
CREATE TABLE GOLD.FACT_TRACK_SALES (
    SALE_ID NUMBER(38,0) NOT NULL,
    INVOICELINE_ID NUMBER(38,0) NOT NULL,
    INVOICEID NUMBER(38,0) NOT NULL,
    TRACKID NUMBER(38,0) NOT NULL,
    TOTAL_REVENUE FLOAT,
    SALE_DATE DATE,
    PRIMARY KEY (SALE_ID)
);
```

Insightful revenue analysis through track metrics becomes possible due to the fact table structure.

Example Analytical Query Using the Gold Layer:

```
SELECT TRACK_NAME, SUM(TOTAL_REVENUE) AS REVENUE
FROM GOLD.FACT_TRACK_SALES
JOIN SILVER.DIM_TRACK ON FACT_TRACK_SALES.TRACKID =
DIM_TRACK.TRACKID
GROUP BY TRACK_NAME
ORDER BY REVENUE DESC
LIMIT 5;
```

The system accesses the five tracks that generate maximum revenue by making use of the optimized Gold Layer structure.

The data architecture that separates information into Bronze Silver and Gold layers creates a data warehouse that simultaneously achieves scalability security and efficiency with optimized quality and performance.

---

2. Integrating Azure Data Lake Storage with Snowflake

Overview

The combination between Azure Data Lake Storage (ADLS) and Snowflake enables secure data ingestion which results in Snowflake accessing structured as well as semi-structured data located directly in Azure Blob Storage. This integration provides:

The system offers data storage scalability which enables efficient handling of extensive data volume.

- Security: Secure authentication via Storage Integrations.
- Cost management optimization becomes possible through dividing storage from compute resources into separate units.
- The query engine of Snowflake processes data directly from Azure without having to perform extra data movement.

2.1 Setting Up the Integration

To connect Snowflake with Azure Data Lake Storage the process requires standard step-by-step execution.

Step 1: Create a Storage Integration in Snowflake

This integration allows Snowflake to securely access data stored in Azure Blob Storage.

```
CREATE OR REPLACE STORAGE INTEGRATION azure_musicplaylist_integration

 TYPE = EXTERNAL_STAGE

 STORAGE_PROVIDER = AZURE

 AZURE_TENANT_ID = '<your-azure-tenant-id>'

 STORAGE_ALLOWED_LOCATIONS =
('azure://musicplaylist.blob.core.windows.net/landing/')

 ENABLED = TRUE;
```

This command:

The documentation sets up an external storage connection to Azure together with an authentication requirement for the Azure Tenant ID.

The configuration allows Azure authentication by specifying a Tenant ID.

The service grants restricted access to certain storage areas for increased safety measures.

Step 2: Create an External Stage in Snowflake

The external stage is developed after the storage integration process enables access to Azure Blob Storage files.

```
CREATE OR REPLACE STAGE adls_stage
```

```
STORAGE_INTEGRATION = azure_musicplaylist_integration
URL = 'azure://musicplaylist.blob.core.windows.net/landing';
```

This command:

The code defines adls_stage as a stage for Azure Blob Storage management.

The coding process uses the already existing azure_musicplaylist_integration for authentication purposes.

The location of data storage in Azure Blob Storage is defined by a specified URL.

Step 3: Define a File Format in Snowflake

The proper loading of structured data requires defining a file format which exactly matches the data structure in incoming files.

```
CREATE OR REPLACE FILE FORMAT csv_file_format
 TYPE = 'CSV'
 FIELD_OPTIONALLY_ENCLOSED_BY = '"'
 SKIP_HEADER = 1;
```

This verification process guarantees correct interpretation of CSV files that reside in the Azure storage system.

2.2 Loading Data from Azure into Snowflake

The next step involves data transfer from Azure Blob Storage into Snowflake tables through already configured storage integration and stage.

Step 4: Copy Data into Snowflake Tables

```
COPY INTO BRONZE.RAW_TRACK
FROM @adls_stage/Track/Track.csv
FILE_FORMAT = (FORMAT_NAME = 'csv_file_format')
ON_ERROR = 'CONTINUE';
```

This command:

- Loads data from Azure Blob Storage (adls_stage) into the RAW_TRACK table.

- The program uses a predefined CSV file format for executing structured data loading tasks.
- The execution continues without interruptions during the loading operation (ON_ERROR = 'CONTINUE').

Step 5: Verify Data Load

To confirm that data has been ingested correctly, we run:

```
SELECT * FROM BRONZE.RAW_TRACK LIMIT 10;
```

This retrieves sample records to verify the integrity of the ingested data.

2.3 Benefits of Using External Stages in Snowflake

Snowflake integration with Azure reaches the following benefits through the utilization of external stages.

- Included within this solution is a process through which Snowflake performs data operations directly in its original position and avoids moving the entire dataset.
- The data pipelines can operate through automated refresh cycles as part of the scheduled maintenance system.
- The optimized access patterns enable faster query performance to occur.

---

3. Using Streams in Snowflake

(I have only created streams but I have not utilised streams in my project I have just showed how to use and I am explaining below how it would be useful to capture newly added records in to the dataset or tables in real time)

Overview of Streams

Snowflake streams provide Change Data Capture functionality to track modifications which occur in a table over time. Streams serve as a tool for efficient incremental processing because they record inserts and updates and deletes effectively.

How Streams Were Used in This Project

To track changes in the **Bronze Layer** before loading data into the **Silver Layer**, we created a **stream** on the `RAW_TRACK` table:

```
CREATE OR REPLACE STREAM Track_Changes_Stream ON TABLE RAW_TRACK
  APPEND_ONLY = TRUE;
```

This stream captures all new or modified records in `RAW_TRACK`. We then use this stream to insert only the changed data into the **Silver Layer**:

```
INSERT INTO SILVER.DIM_TRACK

SELECT * FROM BRONZE.RAW_TRACK

WHERE METADATA$ACTION = 'INSERT';
```

Advantages of Using Streams

- The system tracks changes with efficiency so it avoids reloading complete tables.
- The system processes fresh added information as well as modified data to enhance operational performance.
- The automated data pipeline system allows live information processing between the different system layers.
- The system guarantees that proper data records will move through each stage of propagation.

Our implementation of streams keeps Silver Layer data current in real-time to reduce unnecessary tasks and produce enhanced efficiency.

---

4. Analytical Queries & Insights

Top 5 Tracks with Highest Revenue

```
SELECT

   t.NAME AS TRACK_NAME,

   SUM(f.TOTAL_REVENUE) AS TOTAL_REVENUE

FROM MUSIC_PLAYLIST_DB.GOLD.FACT_TRACK_SALES f

JOIN MUSIC_PLAYLIST_DB.SILVER.DIM_TRACK t

   ON f.TRACKID = t.TRACKID

GROUP BY t.NAME

ORDER BY TOTAL_REVENUE DESC

LIMIT 5;
```

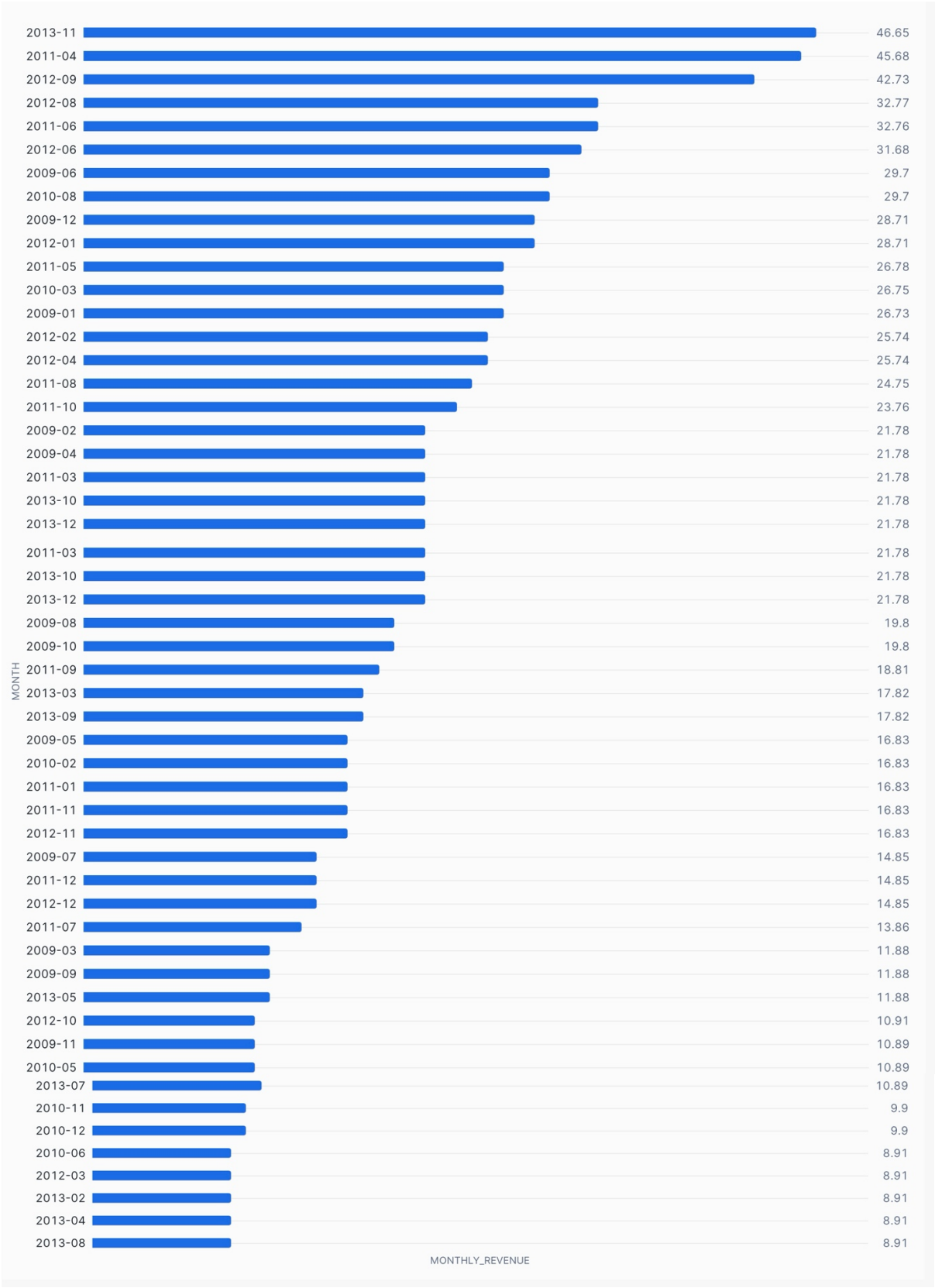
Customers with Highest Lifetime Value (LTV)

```
SELECT
    c.FIRSTNAME || ' ' || c.LASTNAME AS CUSTOMER_NAME,
    SUM(f.TOTAL_SPENT) AS TOTAL_LTV
FROM MUSIC_PLAYLIST_DB.GOLD.FACT_CUSTOMER_PURCHASE_BEHAVIOR f
JOIN MUSIC_PLAYLIST_DB.SILVER.DIM_CUSTOMER c
    ON f.CUSTOMERID = c.CUSTOMERID
GROUP BY CUSTOMER_NAME
ORDER BY TOTAL_LTV DESC
LIMIT 5;
```

Monthly Sales Trend Analysis

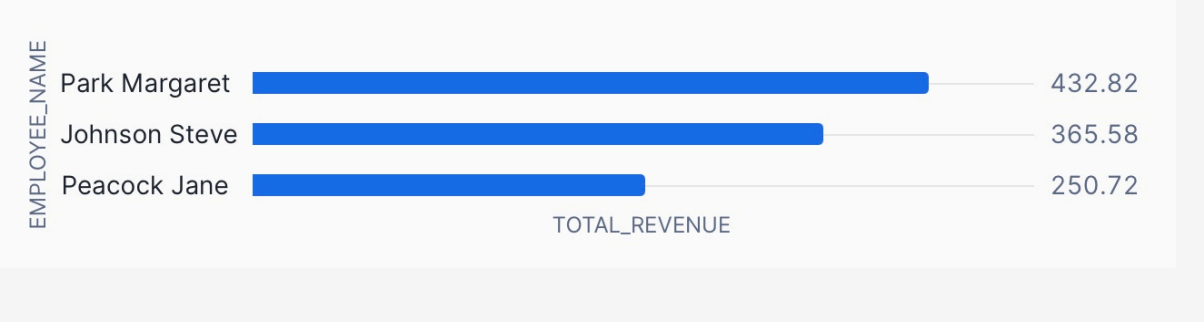
```
SELECT
    TO_CHAR(f.SALE_DATE, 'YYYY-MM') AS MONTH,
```

```
    SUM(f.TOTAL_REVENUE) AS MONTHLY_REVENUE
FROM MUSIC_PLAYLIST_DB.GOLD.FACT_TRACK_SALES f
GROUP BY TO_CHAR(f.SALE_DATE, 'YYYY-MM')
ORDER BY MONTH;
```


2013-11 46.65
2011-04 45.68
2012-09 42.73
2012-08 32.77
2011-06 32.76
2012-06 31.68
2009-06 29.7
2010-08 29.7
2009-12 28.71
2012-01 28.71
2011-05 26.78
2010-03 26.75
2009-01 26.73
2012-02 25.74
2012-04 25.74
2011-08 24.75
2011-10 23.76
2009-02 21.78
2009-04 21.78
2011-03 21.78
2013-10 21.78
2013-12 21.78
2011-03 21.78
2013-10 21.78
2013-12 21.78
2009-08 19.8
2009-10 19.8
2011-09 18.81
2013-03 17.82
2013-09 17.82
2009-05 16.83
2010-02 16.83
2011-01 16.83
2011-11 16.83
2012-11 16.83
2009-07 14.85
2011-12 14.85
2012-12 14.85
2011-07 13.86
2009-03 11.88
2009-09 11.88
2013-05 11.88
2012-10 10.91
2009-11 10.89
2010-05 10.89
2013-07 10.89
2010-11 9.9
2010-12 9.9
2010-06 8.91
2012-03 8.91
2013-02 8.91
2013-04 8.91
2013-08 8.91
MONTH
MONTHLY_REVENUE

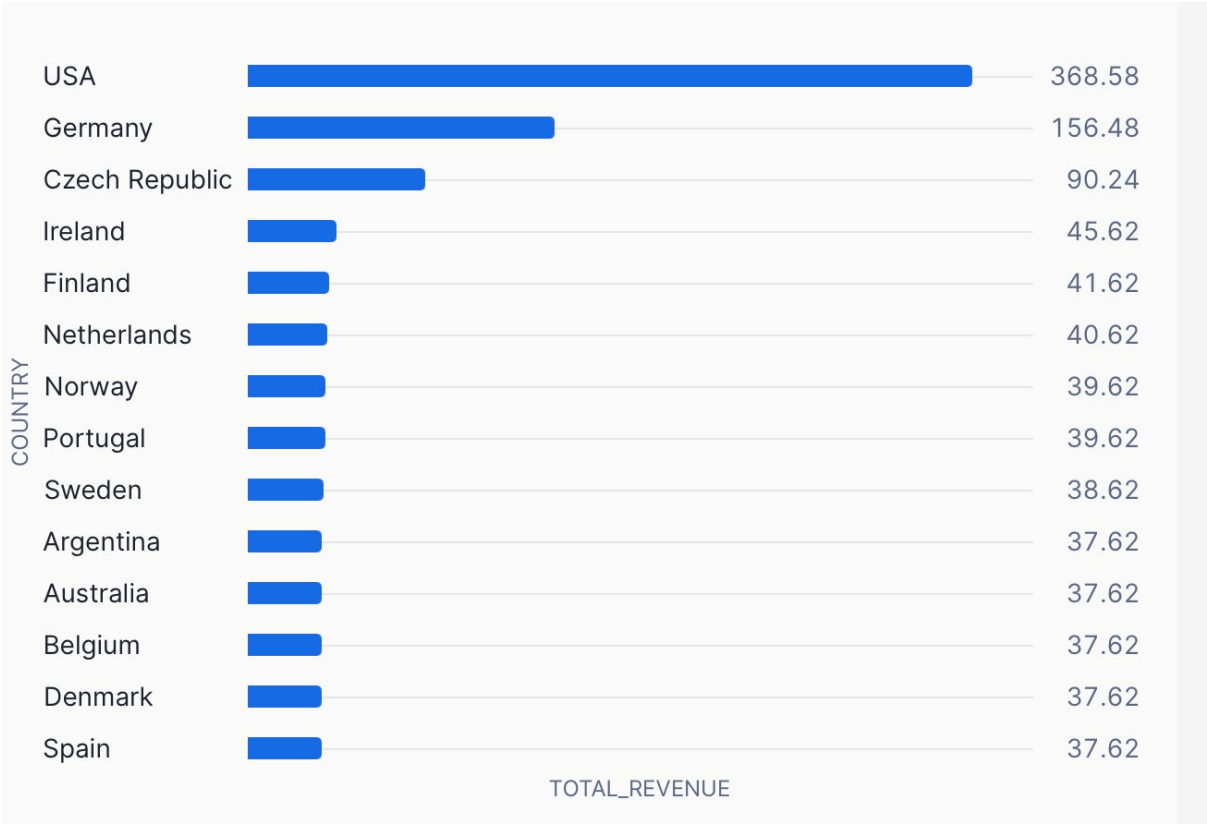
Top 3 Employees by Sales Performance

```
SELECT
    e.FIRSTNAME || ' ' || e.LASTNAME AS EMPLOYEE_NAME,
    SUM(f.TOTAL_SALES) AS TOTAL_REVENUE_GENERATED
FROM MUSIC_PLAYLIST_DB.GOLD.FACT_EMPLOYEE_SALES f
JOIN MUSIC_PLAYLIST_DB.SILVER.DIM_EMPLOYEES e
    ON f.EMPLOYEEID = e.EMPLOYEEID
GROUP BY EMPLOYEE_NAME
ORDER BY TOTAL_REVENUE_GENERATED DESC
LIMIT 3;
```



Country-Wise Revenue Distribution

```
SELECT
    c.COUNTRY,
    SUM(f.INVOICE_AMOUNT) AS TOTAL_REVENUE
FROM MUSIC_PLAYLIST_DB.GOLD.FACT_GEOGRAPHIC_SALES f
JOIN MUSIC_PLAYLIST_DB.SILVER.DIM_CUSTOMER c
    ON f.CUSTOMERID = c.CUSTOMERID
GROUP BY c.COUNTRY
ORDER BY TOTAL_REVENUE DESC;
```



---

5. Benefits of Using Snowflake with Azure

5.1 Advantages of Snowflake

Snowflake functions as a cloud-native data platform which delivers various advantages that establish it as the optimal choice for present-day data warehousing and analytics. The main advantages of Snowflake include:

- The platform enables independent scalability between storage and computing capabilities since it differs from conventional databases which allows users to optimize both price and performance.
- Snowflake uses automatic performance optimization to handle indexing partitioning and query optimization tasks that require no human contact for management.
- The system supports JSON along with Parquet ORC and Avro formats through its native capability to handle semi-structured data.
- This solution provides time travel features with cloning capabilities that let users restore data history and duplicate systems independently.
- Snowflake provides elastic concurrency features through its distributed cluster infrastructure which enables different users to query data at the same time without causing reductions in performance speed.
- Data security receives two layers of protection through encryption models and multi-factor authentication together with role-based access control RBAC mechanism.

5.2 Benefits of Snowflake-Azure Integration

The Azure integration with Snowflake enhances data management and security together with cost-effectiveness capabilities. Here are the key advantages:

1. Seamless Data Ingestion & Integration

- External Stages from Azure Data Lake Storage (ADLS) provide a direct connection using this system.
- The tool allows both automatic and scheduled data transfer operations between Azure and Snowflake.
- This setup removes the requirement for complex ETL pipelines since it enables Azure Blob Storage queries.

2. High Performance & Scalability

- Snowflake operates auto-scaling compute clusters which deals with large workloads at peak times effectively while maintaining performance quality.
- Azure provides its users with access to a global infrastructure that delivers data with low latency and continuous availability.
- The system allows processing of both structured and semi-structured data without needing extensive preprocessing activities.

3. Enhanced Security & Compliance

- End-to-end encryption creates a secure environment for both data storage as well as transfer operations.
- Role-Based Access Control (RBAC) performs access restrictions when user permissions act as the basis for security control decisions.
- The platform operates within the parameters of enterprise security standards GDPR, HIPAA as well as SOC 2.

4. Cost Optimization & Flexible Pricing

- The pay-as-you-go pricing strategy present in Snowflake and Azure helps businesses decrease their infrastructure costs.
- The auto-suspend function of Snowflake activates to deactivate unused computational assets and lessen operational expenses.
- Azure offers treasury storage rates to achieve optimized costs for data storage and retrieval operations.

5. Advanced Analytics & AI Integration

- This platform facilitates integration with Azure Machine Learning as well as Power BI and Databricks to generate AI-based insights.
- Through the connection between Azure Event Hub and Snowflake organizations can achieve real-time analytics capabilities.
- Multiple cloud environments can be queried using a single query command in Snowflake.

5.3 Use Cases for Snowflake-Azure Integration

The complete capabilities of Snowflake blended with Azure offer organizations a perfect solution for many different enterprise needs:

- The platform provides real-time analytics solutions for processing big streams of data which originate from IoT devices together with web applications.
- Financial Data Processing: Handles high-volume transactions securely and efficiently.
- Storms and Azure create an optimal solution which unifies multiple source data to generate personalized customer insights through their Customer Data Platforms.
- Real-time system log monitoring through Log & Security Analysis function allows security threat detection and anomaly identification.
- Healthcare & Life Sciences: Manages large datasets for patient records, genomic research, and clinical analytics.

---

6. Conclusion

The introduction of Music Playlist Database through Snowflake and Azure proved the productivity of extensive data warehouse systems for large music datasets. The structured data processing system of Bronze, Silver and Gold architecture ensures better quality and consistent data and improved access.

The collaboration with Azure Data Lake Storage (ADLS) streamlines data entry whereas Snowflake's advanced features including autonomic scaling together with time rotations and performance optimization delivers optimized data analysis. The data governance and compliance objectives receive increased support through the role-based security model.

The project demonstrated the productive features of cost-efficient data warehouses built on cloud technology which deliver scalable capabilities for business intelligence and analytics to the music industry.

Future Enhancements:

- The team should use machine learning models to analyze Gold-layer tables for extracting additional insights from business data.
- developer should construct interactive visualizations with Power BI to present real-time data.
- Real-time data processing and trend analysis can be accomplished through Azure Event Hub streaming analytics functions.
- The system should expand its database through API integration with third-party music platforms in order to access more detailed information.

With these enhancements, the Music Playlist Database will continue to evolve, providing more value to data analysts and business users by offering actionable insights and advanced reporting capabilities.
