Gold Level Table

Tableau

Purpose

Prepare a gold-level table with over 100 million records to Tableau, available to its users for interactive dashboards, while ensuring:

- Performance (fast) 

- Scalability (potential to grow) 
- Maintainability 

Dataset

The final dataset comes from multiple data sources (final_df):

- casinodaily,
- casinomanufacturers,
- casinoproviders,
- currencyrates,
- users

Contain over 100 million rows

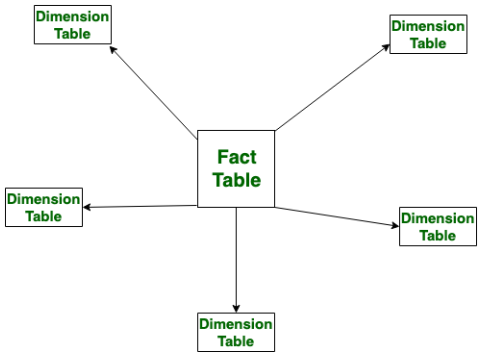
Column Names

- Date,
- Country,
- Sex,
- AgeGroup,
- VIPStatus,
- CasinoManufacturerName,
- CasinoProviderName,
- GGR, and
- Returns

Proposal

Data Storage - Data Warehouse Architecture

- Implement a star schema in a cloud data warehouse (for instance, Databricks, Snowflake), Partitioning & Performance Optimization Strategies



✓ Reason: These platforms are designed for large-scale datasets (>100M records)

✓ How:

- ❑ **Fact Table:** Contains the core metrics (GGR, Returns) with foreign keys to dimension tables
- ❑ **Dimension Tables**: For Date, Country, Sex, AgeGroup, VIPStatus, CasinoManufacturer,CasinoProvider
- ❑ Benefits: Optimized for analytical queries, reduces data duplication
- ❑ **Partitioning on the Date column** (for instance, monthly or even daily) allows the warehouse to scan only the relevant date range when Tableau issues a query.
- ❑ **Create pre-aggregated tables** for commonly used metrics (for instance total GGR by Country)
- ❑ Refresh only new or changed data instead all records.

Tableau Data Connection

- Connect Tableau to the data warehouse via a live connection or extract-based connection, depending on performance and freshness requirements.

1. Live Connection:
   - Pros: Real-time data access, ideal for near-real-time dashboards.
   - Cons: Higher query load on the data warehouse, potentially slower for complex dashboards with many users & very costly.
   - Use Case: Best for small, targeted dashboards with frequent updates
2. Extract-Based Connection:
   - Pros: Pre-computes data into Tableau's Hyper format, enabling faster dashboard performance and offline access. Reduces load on the data warehouse.
   - Cons: Data is not real-time, we set refresh schedules (for instance daily).
   - Use Case: Best for static reporting or when performance is critical for a large user base.

- My recommendation is Extract based Connection with **schedules daily/monthly/hourly** based on business needs. In my experience we have tried to make Live-Connection and they were very costly. They should be suggested only for dashboards with small result sets.

Monitoring and Maintenance & Scaling Considerations

Monitoring and Maintenance

- Implement query performance monitoring
- Set up alerts for long-running queries
- Regularly review and optimize data model as usage patterns emerge.

Scaling Considerations

- Data Growth: Implement data retention policies (archive older than 3 years to cold storage)
- User Growth: Monitor concurrent users and scale warehouse accordingly
- Query Patterns: Create additional aggregated tables for common analytical patterns

Trade-Offs

➢ Performance vs Freshness

- More frequent refreshes impact ETL resources but provide fresher data
- Live connections ensure freshness but may degrade performance with many concurrent users or complex queries
- Solution: Refresh strategy based on dashboard importance

➢ Complexity vs Scalability

- A cloud data warehouse simplifies scaling but requires upfront setup (e.g., partitioning, indexing). Tools like dbt reduce transformation complexity.
- Pandas-based ETL is simple but not scalable for 100M+ records. Moving to SQL-based ELT reduces complexity at scale.

➢ Storage vs Compute

- Materialized views consume storage but reduce compute costs

Future Growth

- **Data Volume**
  The data warehouse can scale to handle growth beyond 100M records. Snowflake/BigQuery automatically scales storage.

- **User Growth**
  Tableau Server/Cloud supports thousands of concurrent users. We use Tableau’s usage analytics to monitor and optimize performance.

- **New Metrics**
  The Gold-level table’s schema (with GGR, Returns) is flexible for new metrics. We can add columns via dbt models without disrupting existing dashboards.