Finpro Bank

Data Architecture, Implementation and Recommendations

Outline

- Introduction
- Data acquisition and cleaning
- Database design and setup
- Evaluation and interpretation
- Data integration and security
- Data reporting
- General handover documentation
- Insights and recommendations
- Summary
- Appendix

Introduction

In today's digital landscape, organizations like FinPro Bank process vast amounts of sensitive customer data, including personal information, account details, and transaction records.

Effective data management is not just a business necessity; it's a regulatory requirement. With the growing complexities of financial data, the bank must implement a robust framework.

Purpose of the presentation

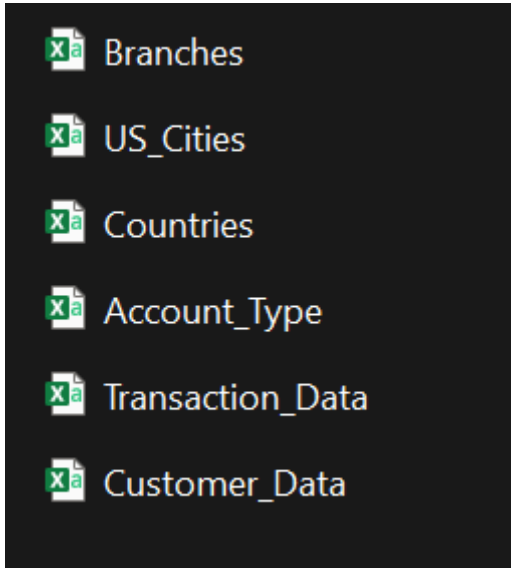
- Outline the project goals, scope, and significance of data-driven decision-making at FinPro Bank.
- Explain the methods used to collect and preprocess data, ensuring accuracy and reliability.
- Showcase the structured database implementation to support efficient data storage and retrieval.
- Discuss how data was analyzed to extract meaningful insights for business operations.
- Highlight the measures taken to protect sensitive data and ensure seamless integration across systems.
- Demonstrate the reporting framework that enables performance monitoring of key performance indicators (KPIs).
- Provide guidelines and documentation to ensure smooth transition and future maintainability.
- Present key findings and actionable steps to optimize FinPro Bank’s data strategy for long-term growth.

Summary of work done on the project

- **Cleaned data**
  - CSVs and Excel
- **Database design and setup**
  - OLTP and OLAP
  - SQL analysis
- **Evaluation and interpretation**
  - Excel reports
- **Data integrity and security**
  - Database privileges, RBAC
- **Data reporting**
  - Tableu dashboard
- **General handover documentation**
  - Strategy recommendations
- **Insights and recommendation**
  - Tableu Story

Data acquisition and cleaning

Data cleaning documentation (1 of 3)



- Source : Data management course (Module 1)
- Format : CSV
- Count : 6
- File sizes : 10- 16 KB

- Transaction data : 6 columns, 21 rows
- Customer data : 10 columns, 21 rows
- Account type : 2 columns, 8 rows
- Countries :4 columns, 192 rows
- Branches : 2 columns, 31 rows
- US_Cities : 3 columns, 127 rows

Data cleaning documentation (2 of 3)

5/21/1990	5	21	1990	21/05/1990
8/15/1985	8	15	1985	15/08/1985
11/3/1992	11	3	1992	03/11/1992
2/28/1978	2	28	1978	28/02/1978
4/7/1989	4	7	1989	07/04/1989
12/13/1975	12	13	1975	13/12/1975
3/20/1993	3	20	1993	20/03/1993
9/15/1987	9	15	1987	15/09/1987

Email
alice.j@example.com
bob.smith@xyz.com
cathy.davis@example.com
david.l@xyz.com
frank.zhang@xyz.com
gina.k@example.com
harry.b@xyz.com
ivy.scott@example.com
karen.g@example.com
liam.m@xyz.com
mona.b@example.com

Balance
12000
3000
8500
5000
-1000
7500
20000
3500
8000
not available

Country
US
United States
US
US
United States
U.S.
Canada
US
Canada
United States
U.S.A.
United Kingdom
US

1. Removing duplicates on IDs.
2. Filling null cells for column email to “email not provided”.
3. Transforming DOB and date related values
    1. Breaking down text to Month-Date-Year using Text to Columns data tools.
    2. Making date using formula “=DATE(year, month, day)”
4. Normalizing country names such as U.S, U.K, etc. to standardized country names in “countries” file data.
5. Transforming and normalizing columns (amount and balance) to numerical datatype
    1. Non numerical changed to 0.

Data cleaning documentation (3 of 3)

Custom	BalanceCategory	AgeGro
34	High Balance	26-35
39	Low Balance	36-50
32	Medium Balance	26-35
47	Medium Balance	36-50
35	Low Balance	26-35
49	Medium Balance	36-50
31	High Balance	26-35
37	Low Balance	36-50
125	Medium Balance	51+
35	Low Balance	26-35

Creating new columns

Customer age using

=DATEDIF(D2; TODAY(); "Y")

Balance Category

=IF(J2>=10000; "High Balance"; IF(J2>=5000; "Medium Balance"; "Low Balance"))

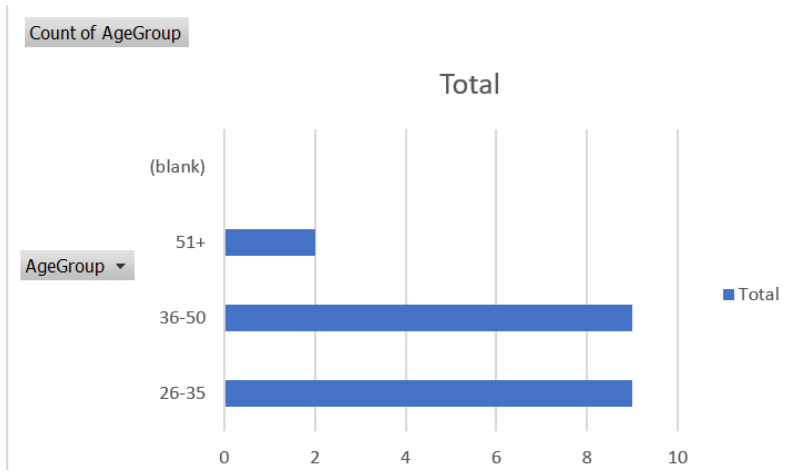
Balance Category

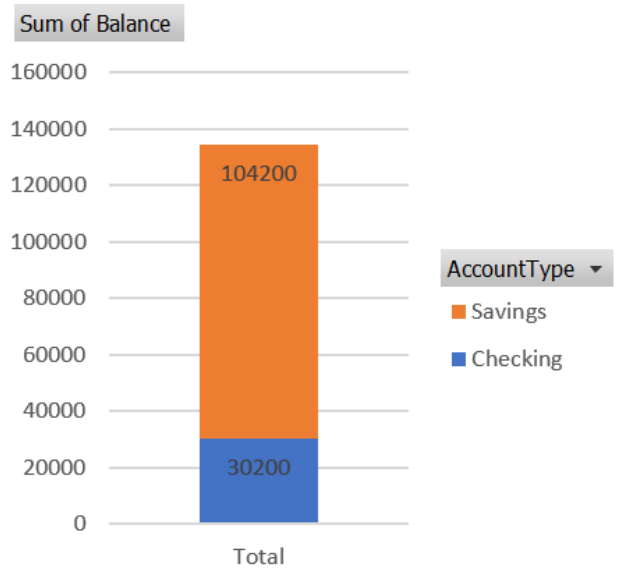
=IFS(K2>=51; "51+"; K2>=36; "36-50"; K2>=26; "26-35"; K2>=18; "18-25")

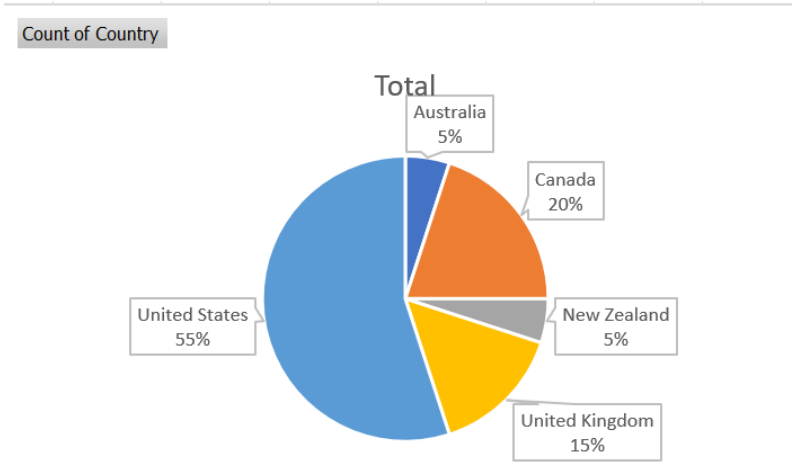
Analysis and visualization using MS Excel (1 of 3)

- KPI Metrics
  - Distribution of age group
  - Sum of balance by account type
  - Customer country
  - Count of transaction in a year
  - Average transaction amount in a year
  - Count of transaction type
- To achieve these KPI metrics visualization, using help of pivot table and then making pivot chart and configure its fields

Analysis and visualization using MS Excel (2 of 3)

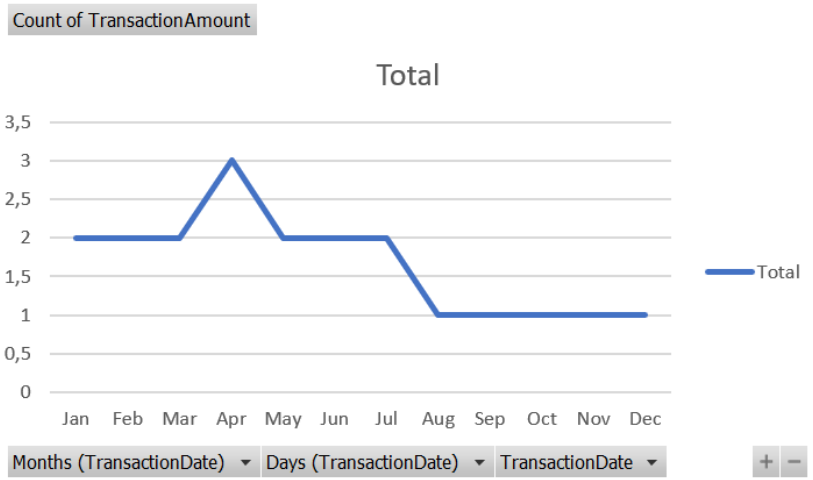


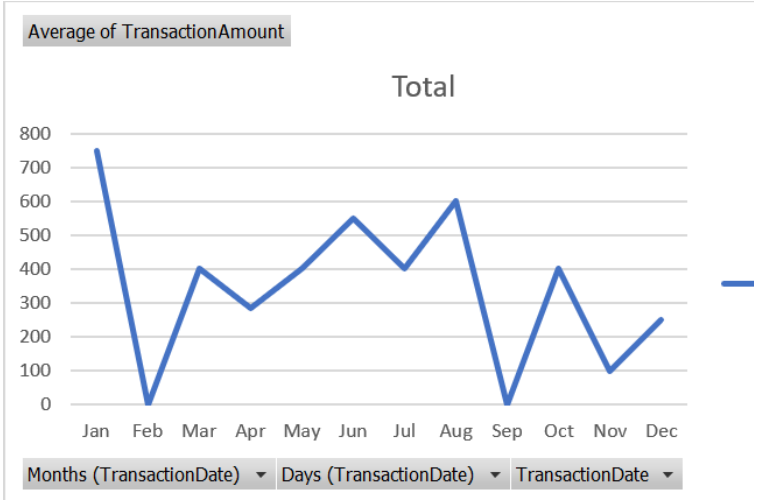


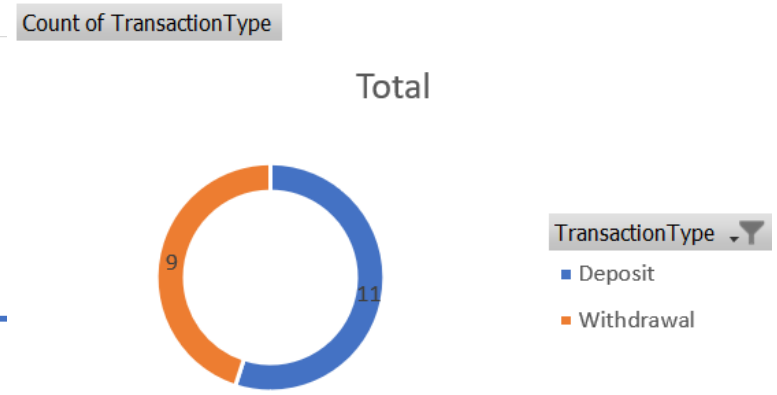


- Customer Age mainly from productive age and early retire (26-50)
- Sum of Balance for account type savings is much higher than checking
- Customer origin country mostly from the United States

Analysis and visualization using MS Excel (3 of 3)



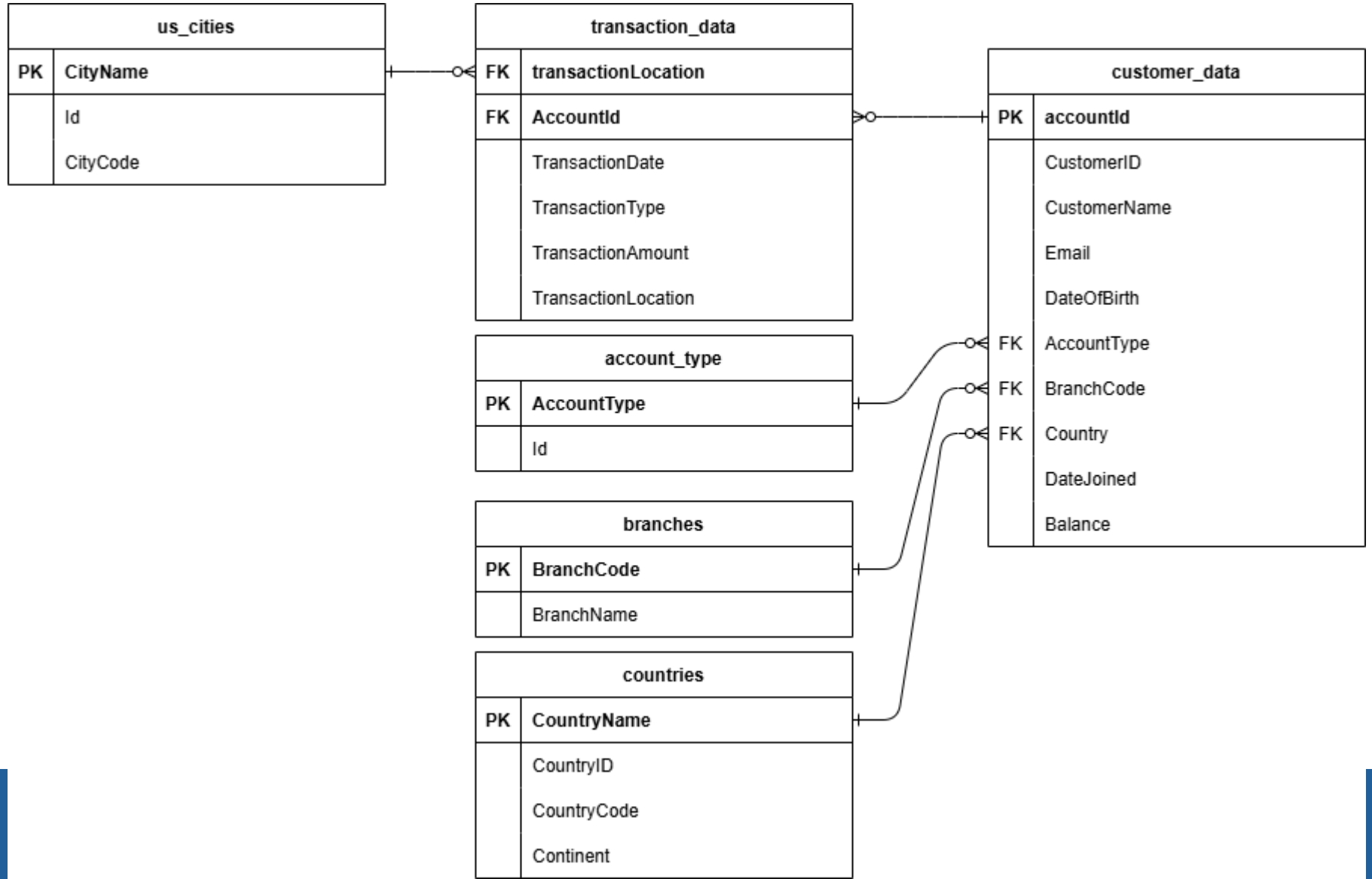




- In 2022, the count of transactions was highest in April and lowest between August and December
- The total average transaction amount was highest in January, followed by a sharp decline in February. It then gradually increased until August before dropping again in September
- The most common transaction type is deposit.

Database design and setup

Entity relationship diagram (ERD) (1 of 3)



Entity relationship diagram (ERD) (2 of 3)

- There are 6 entities for OLTP DB Schema
- Us_cities
  - Has 3 attributes, with PK CityName
  - Related to transaction_data (1 city has many transaction data 1-M)
- Transaction_data
  - Has 6 attributes, with composite PK (TransactionLoc and AccountId)
  - Related to us_cities (many transactions have 1 city M-1)
  - Related to customer data (many transactions have 1 account M-1)
- Customer_data
  - Has 10 attributes, with PK accountId
  - Has 3 FK (AccountType, BranchCode, Country. Many to 1 relation)
- Account_type
  - Has 2 attributes, with PK AccountType
- Branches
  - Has 2 attributes, with PK BranchCode
- Country
  - Has 4 attributes, with PK CountryName

Entity relationship diagram (ERD) (3 of 3)

- Database design for OLTP system
- OLTP ensuring for high Create, Read, Update data
- A customer must have an account in a branch and a country.
- Every transaction must be linked to an existing account.
- Transactions affect the customer’s balance.
- Valid account types must be assigned to customers.
- Branches, cities, and countries enforce geographical integrity.

Table structure design

- Customer_data
- Transaction_data
- Branch
- accountType

Column Name	Data Type	Description
`accountId`	`VARCHAR(255)`	Unique identifier for each customer (Primary Key).
`CustomerID`	`VARCHAR(50)`	Customer's identification number.
`CustomerName`	`VARCHAR(100)`	Full name of the customer.
`Email`	`VARCHAR(100)`	Email address of the customer.
`DateOfBirth`	`DATE`	Customer's date of birth.
`AccountType`	`VARCHAR(50)`	Type of customer account (Foreign Key from `account_type`).
`BranchCode`	`VARCHAR(10)`	Code of the branch where the account is registered (FK from `branches`).
`Country`	`VARCHAR(50)`	Country of the customer (FK from `countries`).
`DateJoined`	`DATE`	Date when the customer joined the bank.
`Balance`	`DECIMAL(15,2)`	Current balance of the customer.

Column Name	Data Type	Description
`TransactionID`	`INT` (Auto-Increment)	Unique identifier for each transaction (Primary Key).
`AccountId`	`VARCHAR(255)`	Associated customer account (Foreign Key from `customer_data`).
`TransactionDate`	`DATETIME`	Date and time of the transaction.
`TransactionType`	`VARCHAR(50)`	Type of transaction (Deposit, Withdrawal, Transfer, etc.).
`TransactionAmount`	`DECIMAL(15,2)`	Amount involved in the transaction.
`TransactionLocation`	`VARCHAR(100)`	City where the transaction was made (Foreign Key from `us_cities`).

Column Name	Data Type	Description
`BranchCode`	`VARCHAR(10)`	Unique identifier for each bank branch (Primary Key).
`BranchName`	`VARCHAR(100)`	Name of the branch.

Column Name	Data Type	Description
`AccountType`	`VARCHAR(50)`	Unique type of account (Primary Key).
`Id`	`INT` (Auto-Increment)	Internal reference ID.

Table structure design

- Country
- US_City

Column Name	Data Type	Description
`CountryName`	`VARCHAR(100)`	Name of the country (Primary Key).
`CountryID`	`VARCHAR(10)`	Unique identifier for the country.
`CountryCode`	`VARCHAR(5)`	ISO country code.
`Continent`	`VARCHAR(50)`	Continent where the country is located.

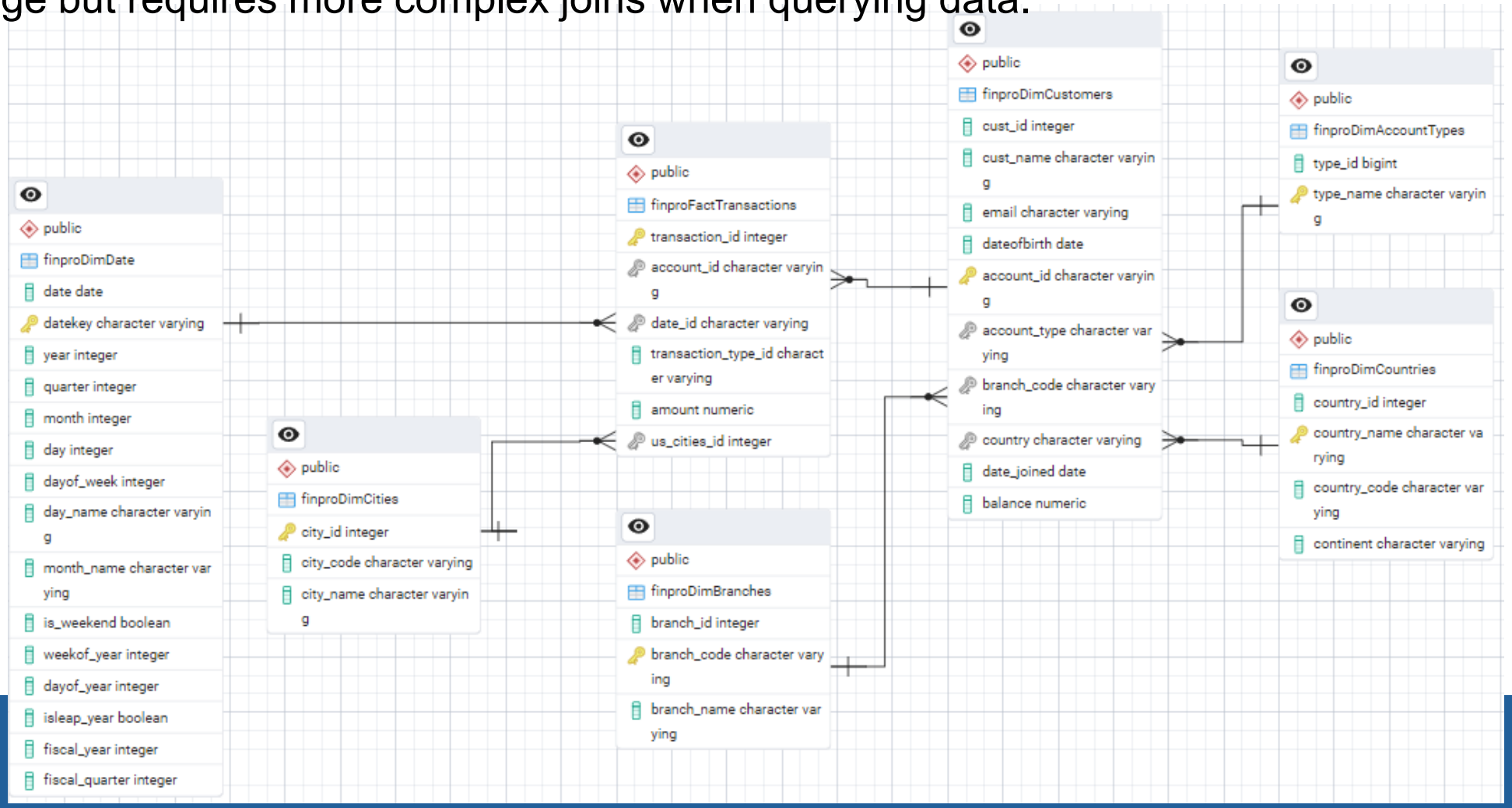
Column Name	Data Type	Description
`CityName`	`VARCHAR(100)`	Name of the city (Primary Key).
`Id`	`INT` (Auto-Increment)	Internal reference ID.
`CityCode`	`VARCHAR(10)`	Unique identifier for the city.

- Relationship

Relationship	Type	Description
`customer_data` ↔ `transaction_data`	1:M	One customer can have multiple transactions.
`customer_data` ↔ `account_type`	M:1	Many customers can have the same account type.
`customer_data` ↔ `branches`	M:1	Many customers belong to a single branch.
`customer_data` ↔ `countries`	M:1	Many customers belong to a single country.
`transaction_data` ↔ `us_cities`	M:1	Many transactions can occur in one city.

OLAP schema (1 of 3)

- SNOWFLAKE SCHEMA
- dimension tables are further normalized into multiple related tables.This design optimizes storage but requires more complex joins when querying data.



OLAP schema (2 of 3)

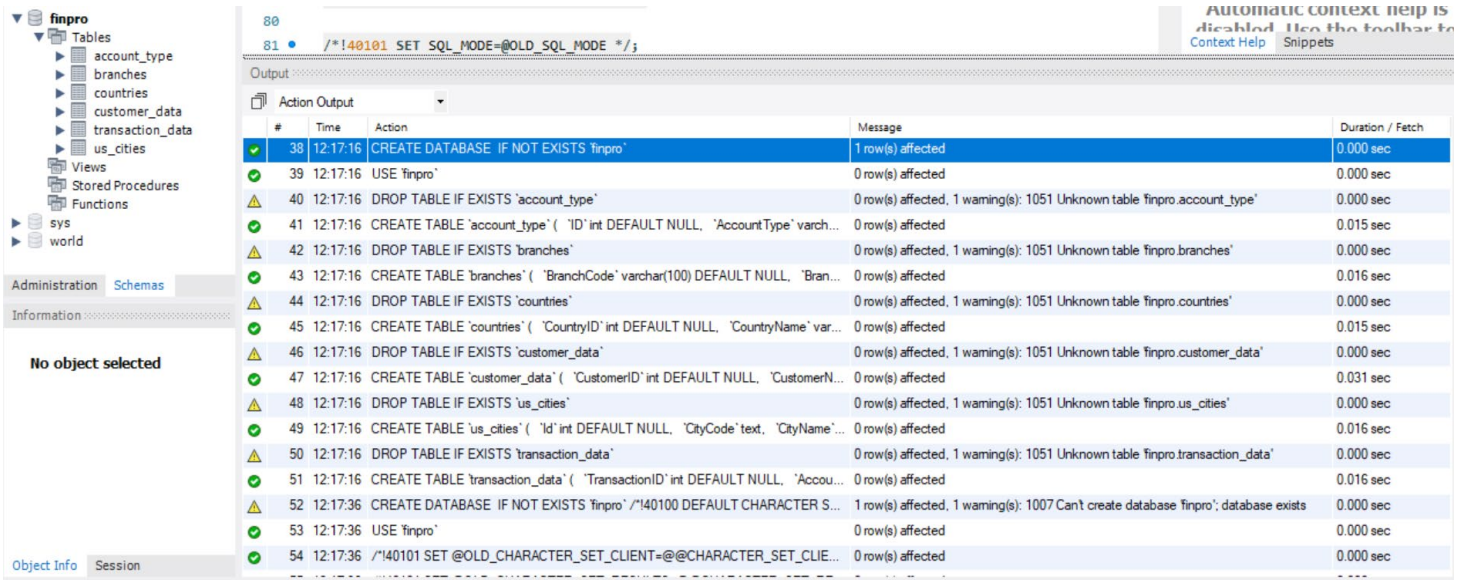
- There are 7 entities for OLAP DB Schema
- **DimDate**
  - Has 15 attributes, with PK dateKey
  - Related to DimTransactions (1 date has many transactions 1:M)
- **DimCities**
  - Has 3 attributes, with PK CityName
  - Related to transaction_data (1 city has many transaction data 1-M)
- **Transactions**
  - Has 6 attributes, with PK transaction_id
  - Related to DimCities (many transactions have 1 city M:1)
  - Related to DimCustomers-accountId (many transactions have 1 account M:1)
- **DimCustomers**
  - Has 10 attributes, with PK accountId
  - Has 3 FK (AccountType, BranchCode, Country. Many to 1 relation)
- **DimAccountTypes**
  - Has 2 attributes, with PK AccountType
- **DimBranches**
  - Has 2 attributes, with PK BranchCode
- **DimCountries**
  - Has 4 attributes, with PK CountryName

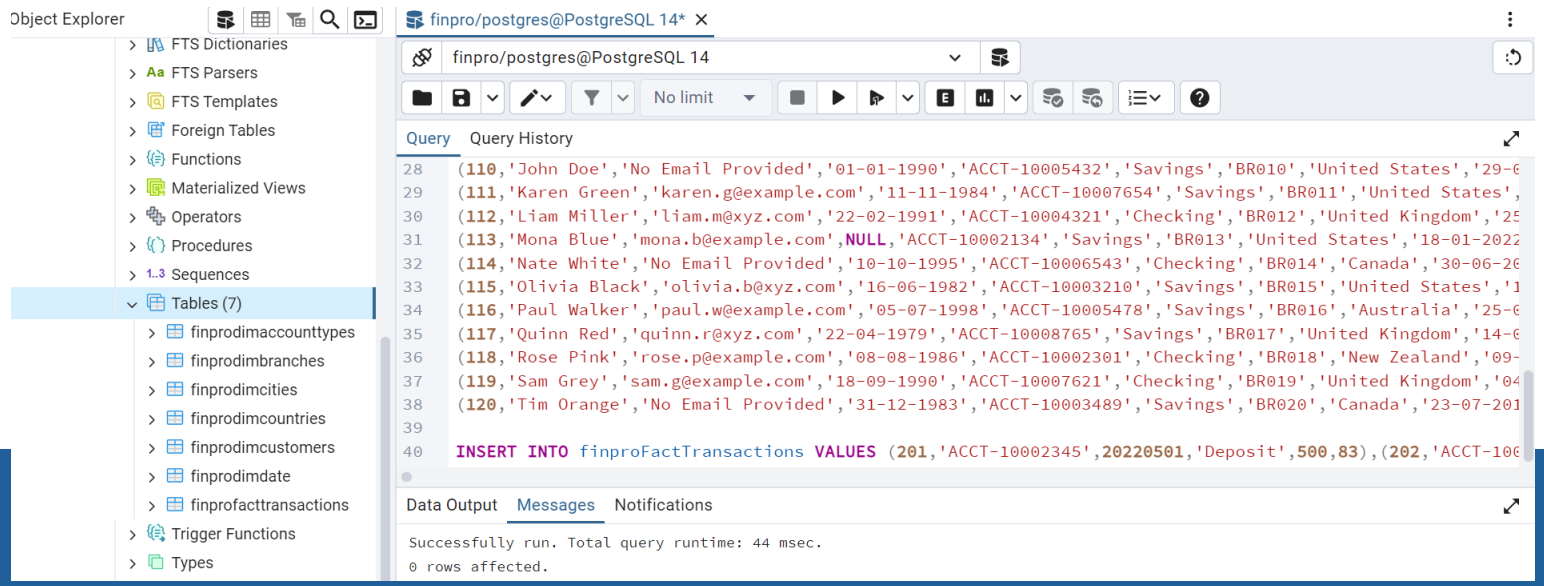
OLAP schema (3 of 3)

- This Star Schema design is optimized for fast query performance analysis
- Business purpose
  - Trend Analysis: Find peak transaction months, customer behaviors.
  - Geographical Insights: Identify high-transaction locations.
  - Customer Segmentation: Analyze account types, balances.
  - Branch Performance: Track which branches generate more transactions.
  - Time-based Analysis: Compare yearly, monthly, or daily transactions.

Evaluation and interpretation

Documenting SQL queries and scripts (1 of 4)





- OLTP successfully implemented
  - Data Definition Languange (DDL)
  - Insert statements
  - Using local MySQL
- OLAP successfully implemented
  - Data Definition Languange (DDL)
  - Insert statements
  - Using PostgreSQl

Documenting SQL queries and scripts (2 of 4)

- Sample Queries for OLTP
- Create
  - INSERT INTO transaction_data (transaction_id, accountId, transactionDate, transactionType, transactionAmount, transactionLocation) VALUES (1, 'ACC123', '2024-03-04', 'Deposit', 1000.00, 'New York');
- Read
  - SELECT * FROM transaction_data WHERE accountId = 'ACC123'
- Update
  - UPDATE transaction_data SET transactionAmount = 1200.00 WHERE transaction_id = 1;
- Delete
  - DELETE FROM branches WHERE branchCode = 'BR001';

Documenting SQL queries and scripts (3 of 4)

Query | Query History

```sql
-- grouping set
-- define multiple grouping levels in one query.
SELECT cust_id, account_id, SUM(balance) AS total_balance
FROM finproDimCustomers
GROUP BY GROUPING SETS (
    (cust_id),
    (account_id),
    (cust_id, account_id),
    ()
);
```

Data Output | Messages | Notifications

	cust_id integer	account_id [PK] character varying	total_balance numeric
1	[null]	[null]	134400
2	116	ACCT-10005478	-500
3	108	ACCT-10006789	3500
4	105	AC	
5	114	AC	
6	119	AC	
7	109	AC	

Total rows: 61 of 61

Query | Query History

```sql
-- Rollup
-- hierarchical aggregations for selected columns
SELECT branch_code, account_type, SUM(amount) AS total_transaction
FROM finproFactTransactions t
JOIN finproDimCustomers c ON t.account_id = c.account_id
GROUP BY ROLLUP (branch_code, account_type);
```

Data Output | Messages | Notifications

	branch_code character varying	account_type character varying	total_transaction numeric
1	[null]	[null]	7200
2	BR002	Checking	200
3	BR019	Checking	350
4	BR016	Savings	500
5	BR015	Savings	-200
6	BR001	Savings	500
7	BR013	Savings	250
8	BR009	Savings	600
9	BR003	Savings	300
10	BR004	Savings	-400

Successfully run. Tota

Total rows: 41 of 41 | Query complete 00:00:00.070 | Ln 16, Col 48

Query | Query History

```sql
-- Cube
-- provides all possible grouping combinations
SELECT
    EXTRACT(YEAR FROM date_joined) AS year_of_joining,
    country,
    SUM(balance + COALESCE(amount, 0)) AS closing_balance
FROM finproDimCustomers c
LEFT JOIN finproFactTransactions t ON c.account_id = t.account_id
GROUP BY CUBE (EXTRACT(YEAR FROM date_joined), country);
```

Data Output | Messages | Notifications

	year_of_joining numeric	country character varying	closing_balance numeric
1	[null]	[null]	141600
2	2022	United States	4250
3	2019	United Kingdom	29400
4	2018	United States	7600
5	2019	Canada	9800
6	2020	United States	15650
7	2022	Canada	20750
8	2021	Canada	8600

Total rows: 24 of 24 | Query complete 00:00:00.065 | Ln 23,

Query | Query History

```sql
-- MQT
-- store query results, improving performance.
CREATE MATERIALIZED VIEW Total_transaction_per_branch AS
SELECT c.branch_code, SUM(t.amount) AS total_transaction
FROM finproFactTransactions t
JOIN finproDimCustomers c ON t.account_id = c.account_id
GROUP BY c.branch_code;

REFRESH MATERIALIZED VIEW Total_transaction_per_branch;
SELECT * FROM Total_transaction_per_branch;
```

Data Output | Messages | Notifications

	branch_code character varying	total_transaction numeric
1	BR003	300
2	BR010	0
3	BR013	250
4	BR019	350
5	BR012	100
6	BR005	450
7	BR018	700

Success

Total rows: 20 of 20 | Query complete 00:00:00.056 | Ln 38, Col 17

Documenting SQL queries and scripts (4 of 4)

Query | Query History

```
41  SELECT
42      t.account_id,
43      SUM(
44          CASE
45              WHEN transaction_type_id IN ('Deposit', 'Credit') THEN amount
46              WHEN transaction_type_id IN ('Withdrawal', 'Debit') THEN -amount
47              ELSE 0
48          END
49      ) AS closing_balance
50  FROM finproFactTransactions t
51  GROUP BY t.account_id
52  ORDER BY closing_balance DESC;
```

Data Output | Messages | Notifications

	account_id character varying	closing_balance numeric
1	ACCT-10006543	1000
2	ACCT-10008765	800
3	ACCT-10008901	750
4	ACCT-10009876	600
5	ACCT-10002345	500

Total rows: 20 of 20 Query complete 00:00:00.060 Ln 43, Col 7

- This query calculates the closing balance for each account by summing up deposits and withdrawals.
- Deposits and credits increase the balance.
- Withdrawals and debits decrease the balance.
- The ORDER BY closing_balance DESC sorts accounts with the highest balance first.

Query | Query History

```
54  SELECT
55      c.cust_id,
56      c.cust_name,
57      COUNT(CASE WHEN t.transaction_type_id IN ('Deposit', 'Credit') THEN 1 END) AS deposit_count,
58      COUNT(CASE WHEN t.transaction_type_id IN ('Withdrawal', 'Debit') THEN 1 END) AS withdrawal_count
59  FROM finproDimCustomers c
60  LEFT JOIN finproFactTransactions t ON c.account_id = t.account_id
61  GROUP BY c.cust_id, c.cust_name
62  ORDER BY deposit_count DESC, withdrawal_count DESC;
63
```

Data Output | Messages | Notifications

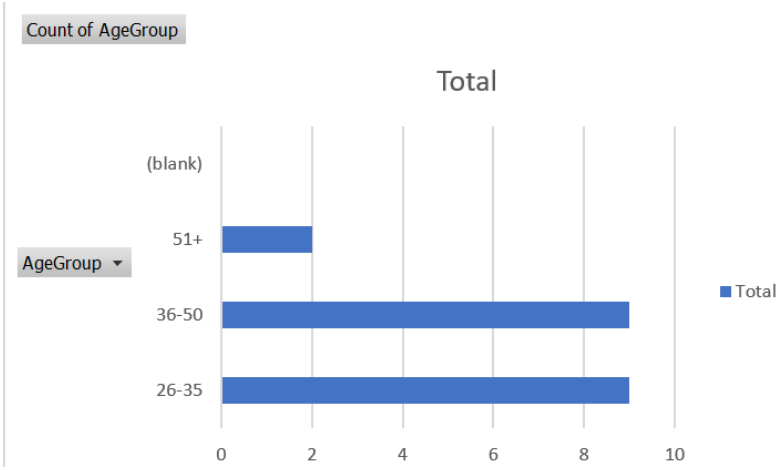
	cust_id integer	cust_name character varying	deposit_count bigint	withdrawal_count bigint
1	114	Nate White	1	0
2	107	Gina King	1	0
3	119	Sam Grey	1	0
4	117	Quinn Red	1	0
5	113	Mona Blue	1	0
6	101	Alice Johnson	1	0

Total rows: 20 of 20 Query complete 00:00:00.076 Ln 58, Col 19

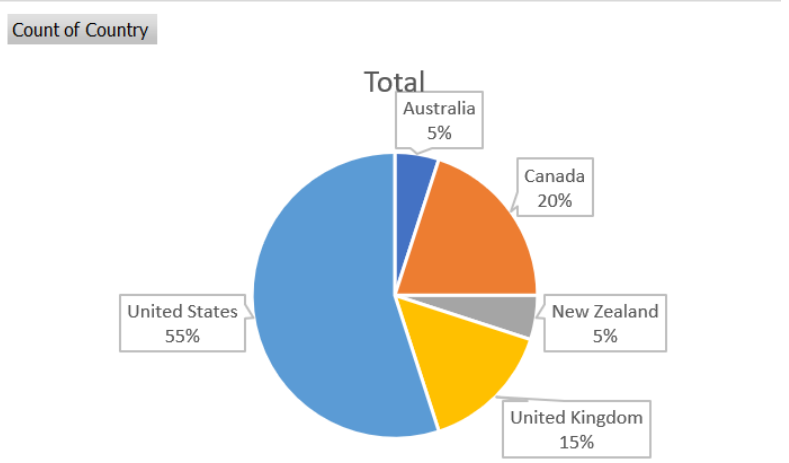
- Counts the number of deposits and withdrawals per customer.
- Uses CASE WHEN inside COUNT() to separately count deposits and withdrawals.
- LEFT JOIN ensures all customers are included, even those with no transactions.
- ORDER BY deposit_count DESC, withdrawal_count DESC sorts customers by most transactions.

Inferences (1 of 2)

- **Inferences from OLTP**
- Insights from the Age Group ChartThe majority of respondents fall in the 26–35 and 36–50 age groups
- These two groups have the highest counts, suggesting that the survey or data source primarily involves middle-aged individuals.
  - Targeting new customers in 26–50 age range may be more effective.
  - Increase participation among older individuals might be needed.



- Insights from the Country Distribution Pie ChartThe
- United States has the highest representation (55%).This suggests that the dataset is heavily skewed towards U.S. respondents, making it the dominant demographic.
- Canada (20%) and the United Kingdom (15%) have moderate representation.These countries make up a significant portion of the data, meaning they can still provide valuable insights.
- If the focus is on the U.S. market, it may already be well-represented, allowing for targeted analysis.
- The low participation from Australia and New Zealand suggests that region-specific outreach efforts could be beneficial.



Inferences (2 of 2)

- **Inferences from OLTP**
- **Overdrawn or Low-Balance Accounts Require Intervention**
  - The Closing Balance Per Account query revealed negative balances for some accounts
  - Actionable: Send alerts to these customers, offering repayment plans or overdraft protection.

	account_id character varying	closing_balance numeric
10	ACCT-10007621	350
11	ACCT-10007891	300
12	ACCT-10002134	250
13	ACCT-10003210	200
14	ACCT-10005432	0
15	ACCT-10004567	-100
16	ACCT-10004321	-100
17	ACCT-10005678	-200
18	ACCT-10006789	-200
19	ACCT-10003489	-600
20	ACCT-10002301	-700

Total rows: 20 of 20 Query complete 00:00:00.066

- **High-Transaction Customers Are Key for Revenue Growth**
  - The Deposit & Withdrawal Counts Per Customer query shows top customers by transaction volume.
  - Actionable: Identify such high-engagement customers for premium banking services, exclusive offers, or priority support.

	cust_id integer	cust_name character varying	deposit_count bigint	withdrawal_count bigint
6	101	Alice Johnson	1	0
7	109	Ivy Scott	1	0
8	111	Karen Green	1	0
9	116	Paul Walker	1	0
10	105	Eva Turner	1	0
11	103	Cathy Davis	1	0
12	104	David Lee	0	1
13	110	John Doe	0	1
14	118	Rose Pink	0	1
15	112	Liam Miller	0	1
16	106	Frank Zhang	0	1

Data integration and security

Data integration recommendations (1 of 3)

- **areas requiring data integration**
- Customer Data Integration
  - Customer information is often scattered across multiple systems (e.g., core banking, CRM, mobile banking). Integrating customer data ensures a unified view of customers for better service, risk assessment, and personalized offerings.
- Transaction Data Synchronization
  - Real-time synchronization of transactions (deposits, withdrawals, transfers) between core banking, ATM networks, and digital banking platforms is necessary to maintain accuracy and prevent overdrafts or fraud.
- Fraud Detection & Risk Management
  - Fraud detection systems need access to real-time transaction feeds to flag suspicious activities instantly.
- Regulatory & Compliance Reporting
  - Banks must comply with financial regulations to ensure compliance.
- Business Intelligence & Analytics
  - A centralized data warehouse integrating customer, transaction, and operational data enables accurate KPI reporting and strategic decision-making

Data integration recommendations (2 of 3)

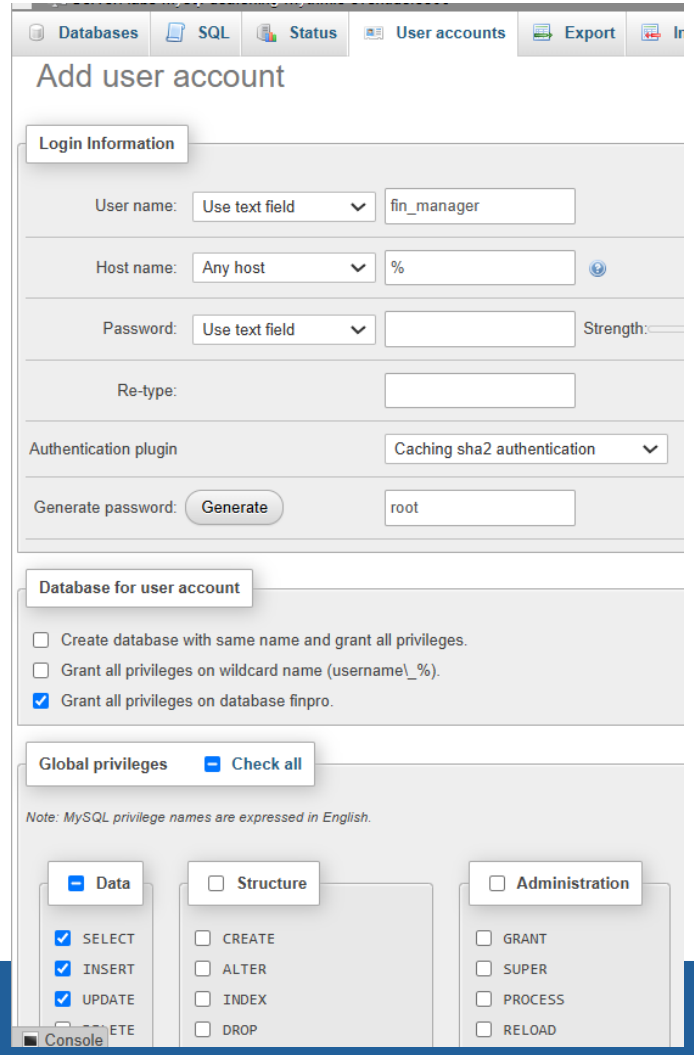
- **Recommended Data Integration**
- ETL (Extract, Transform, Load) for Batch Processing
  - ETL ensures high-quality, structured data is available for BI reporting and analytics.
  - Extracting daily transactions from core banking and loading them into a data warehouse for financial reporting.
- API-Based Data Integration for Open Banking & Digital Services
  - RESTful and GraphQL APIs ensure secure, on-demand access to banking services
  - Ex. account balance checks and transactions
- Event-Driven Streaming
  - Real time data monitoring for Fraud detection systems in real-time
  - Ex. triggers an instant alert for potential fraud.

Data integration recommendations (3 of 3)

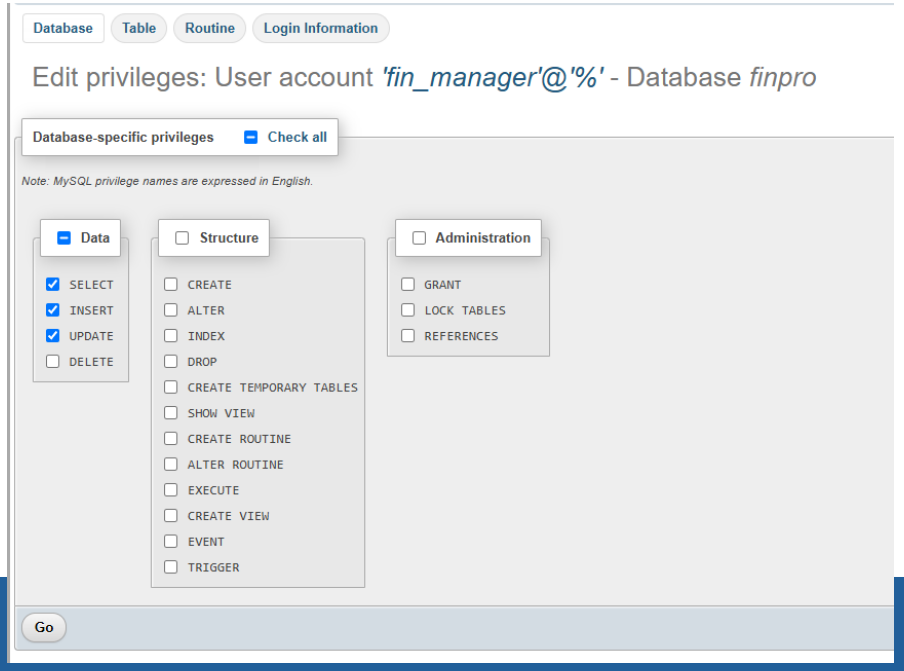
- Real-time & Batch Processing Balance
  - The combination of ETL (for BI & reporting) and event-driven architecture (for fraud detection & live updates)
- API-First Approach for Open Banking
  - With the rise of fintech partnerships and mobile banking, API-based integration ensures secure customer interactions.
- Compliance & Security Prioritization
  - Integration regulatory systems ensures FinPro meets financial regulations.
- Scalability for Growth
  - Event-driven systems and data virtualization support future expansions without major infrastructure changes.

Security documentation (1 of 3)

- Detailed setting RBAC



- Added new user “fin_manager”
- Add global privileges for select, insert, and update
- For database level privileges, check select, insert, and update
- With these steps, fin_Manager can only do selected privileges for finpro database.



Security documentation (2 of 3)

- Detailed setting Encryption

```
7  SET @key_str = SHA2('My secret passphrase', 512);
8  ALTER TABLE customer_data MODIFY COLUMN AccountId varbinary(255);
9  SET SQL_SAFE_UPDATES = 0;
10 UPDATE customer_data  SET AccountId = AES_ENCRYPT(AccountId , @key_str);
11
12 SELECT *, cast(AES_DECRYPT(AccountId , @key_str) as char(255)) FROM customer_data;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

irth	AccountId	AccountType	BranchCode	Country	DateJoined	Balance	cast(AES_DECRYPT(AccountId , @key_str) as char(255))
0	BLOB	Savings	BR001	United States	1/15/2020	12000	ACCT-10002345
5	BLOB	Checking	BR002	United States	11/30/2019	3000	ACCT-10005678
2	BLOB	Savings	BR003	United States	6/1/2021	8500	ACCT-10007891
8	BLOB	Savings	BR004	United States	3/20/2021	5000	ACCT-10001234

- Encrypt accountId column to ensure security
- Setting key and encrypt using SHA2
- Alter table column accountID to VARBINARY to save encrypted values
- Set safe update to 0, forcing update accountId values
- Updating accountID value to encrypted values
- Encrypt success

CustomerID	CustomerName	Email	DateOfBirth	AccountId	AccountType	BranchCode	Country
101	Alice Johnson	alice.j@example.com	5/21/1990	ACCT-10002345	Savings	BR001	United States
102	Bob Smith	bob.smith@xyz.com	8/15/1985	ACCT-10005678	Checking	BR002	United States
103	Cathy Davis	cathy.davis@example.com	11/3/1992	ACCT-10007891	Savings	BR003	United States
104	David Lee	david.l@xyz.com	2/28/1978	ACCT-10001234	Savings	BR004	United States
105	Eva Turner	No Email Provided	4/7/1989	ACCT-10003456	Checking	BR005	United States

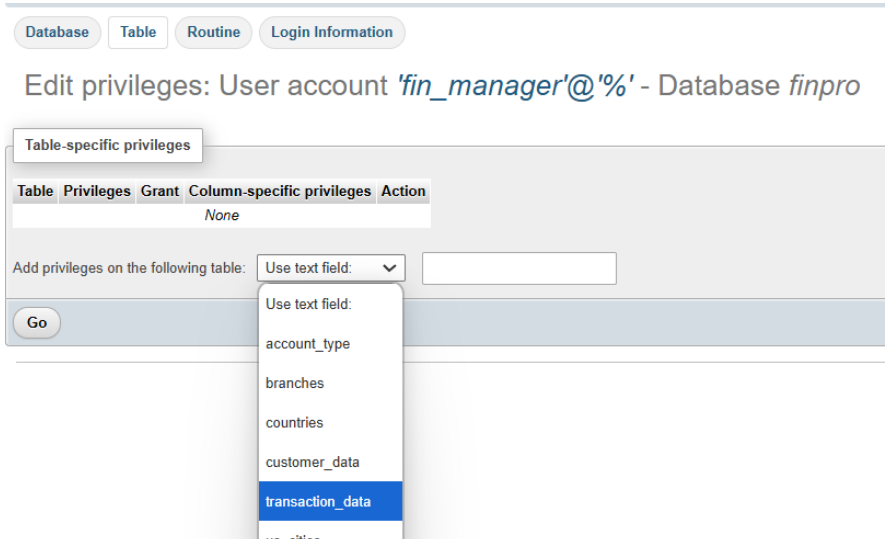
- Figure 1 Implemented query to encrypt and examples to decrypt IDs
- Figure 2 before accountID encrypted (plain sight)
- Figure 3 after accountID encrypted (blob)

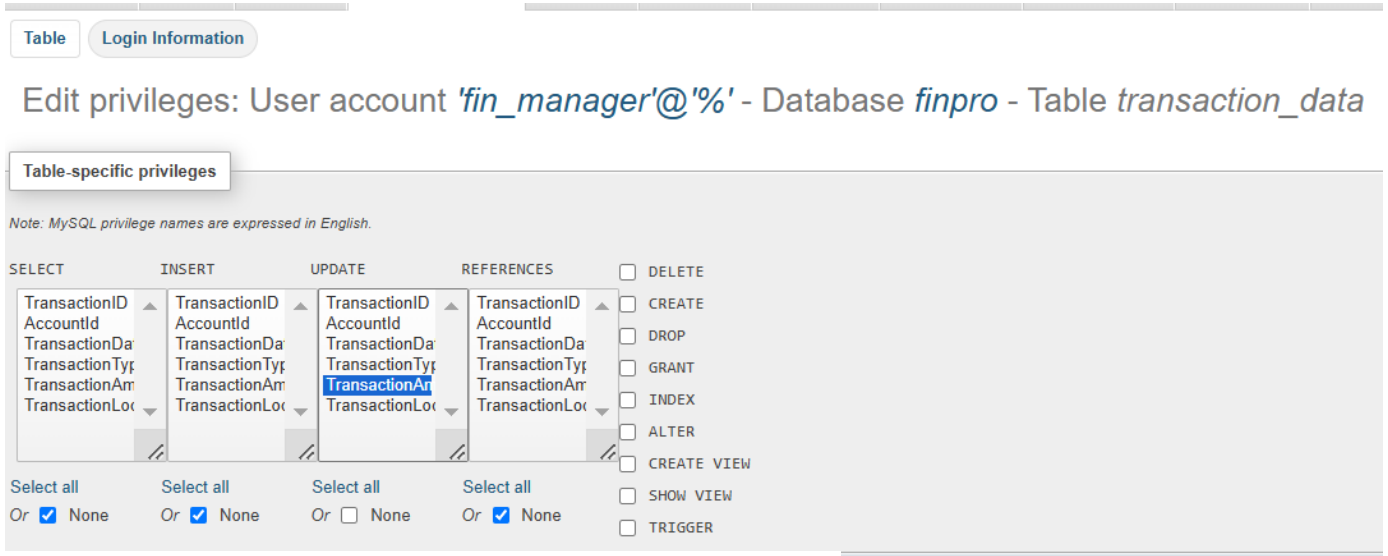
CustomerID	CustomerName	Email	DateOfBirth	AccountId	AccountType	BranchCode	Country	Da
101	Alice Johnson	alice.j@example.com	5/21/1990	BLOB	Savings	BR001	United States	1/1
102	Bob Smith	bob.smith@xyz.com	8/15/1985	BLOB	Checking	BR002	United States	11/
103	Cathy Davis	cathy.davis@example.com	11/3/1992	BLOB	Savings	BR003	United States	6/1,
104	David Lee	david.l@xyz.com	2/28/1978	BLOB	Savings	BR004	United States	3/2
105	Eva Turner	No Email Provided	4/7/1989	BLOB	Checking	BR005	United States	7/1

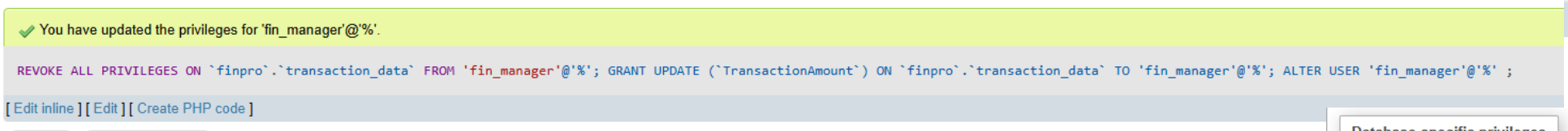
Security documentation (3 of 3)

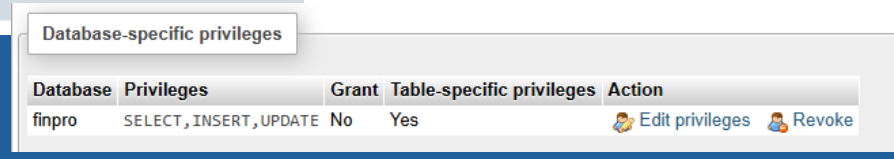
- Table specific privileges

- Follow through (security documentation 1 of 3)
- Setting table level privileges for transaction_data table
- Click none for select, insert, references, in the update section select transactionAmount
- This action restrict fin_manager to select, insert, and update, all columns in transaction_data. Fin_manager can only do update for column transactionAmount





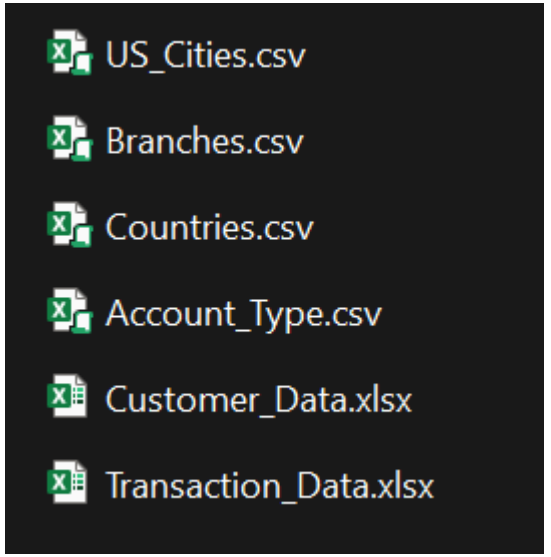




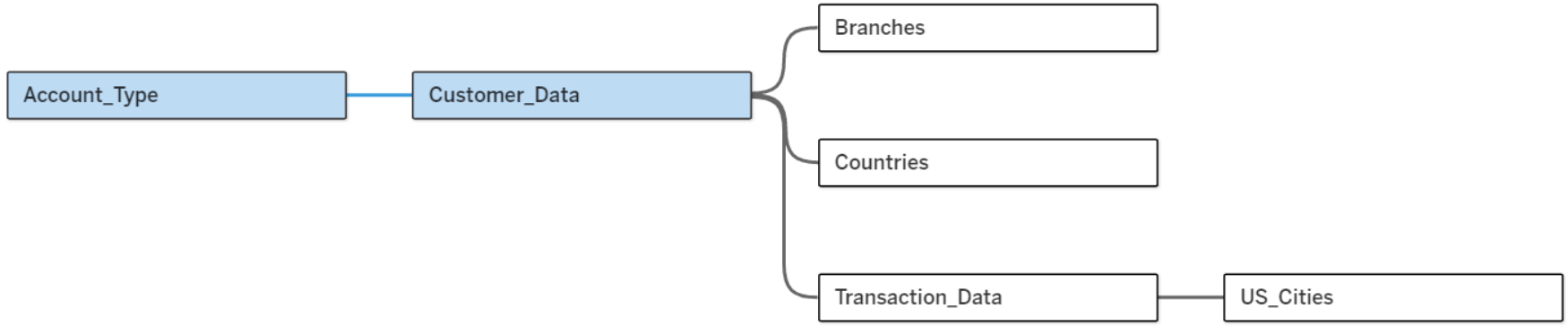
Data reporting

Report specifications (1 of 6)

- Transaction data
  - Contains transactional records, including deposits, withdrawals.
  - Key for financial reporting, closing balance calculations, and fraud detection analysis.
- Customer data
  - Contains customer details such as name, age, country, account type, and joining date.
  - Essential for customer segmentation, age group analysis, and understanding customer demographics.
- Account type
  - Lists different types of bank accounts (e.g., savings, checking, credit).
  - Helps categorize transactions and analyze trends based on account type.
- Countries
  - Contains a list of countries where customers or transactions originate.
  - Used to segment data geographically for demographic insights and compliance reporting.
- Branches
  - Contains data about bank branches, such as branch codes, locations
  - Useful for analyzing branch performance, customer distribution, and transaction volume per branch.
- US_Cities
  - Contains a list of cities
  - Helps in regional analysis of customer distribution, transaction trends, and branch locations.

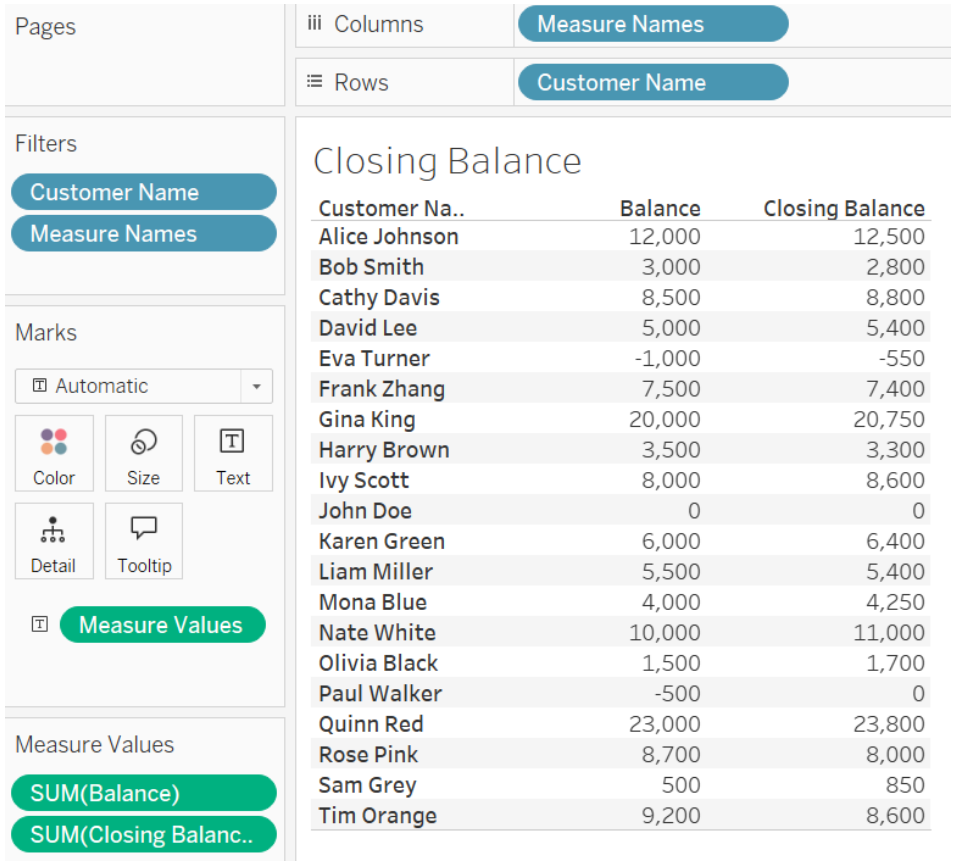


Report specifications (2 of 6)



- Key filters
  - Account type - Helps categorize transactions and analyze trends based on account type.
  - Customer_data - Essential for customer segmentation, understanding customer
  - Branches - Useful for analyzing branch performance
  - Countries - Used to segment data geographically by country
  - Transaction_data - Key for financial reporting, customer behaviour
  - Us_cities - Helps in regional analysis by cities
  - Closing Balance - final amount in each account after all transactions.
  - Age - Segments customers into age groups

Report specifications (3 of 6)



Closing Balance

Customer Na..	Balance	Closing Balance
Alice Johnson	12,000	12,500
Bob Smith	3,000	2,800
Cathy Davis	8,500	8,800
David Lee	5,000	5,400
Eva Turner	-1,000	-550
Frank Zhang	7,500	7,400
Gina King	20,000	20,750
Harry Brown	3,500	3,300
Ivy Scott	8,000	8,600
John Doe	0	0
Karen Green	6,000	6,400
Liam Miller	5,500	5,400
Mona Blue	4,000	4,250
Nate White	10,000	11,000
Olivia Black	1,500	1,700
Paul Walker	-500	0
Quinn Red	23,000	23,800
Rose Pink	8,700	8,000
Sam Grey	500	850
Tim Orange	9,200	8,600

- This Closing Balance Report in Tableau provides an overview of customer balances, showing both their initial Balance and the Closing Balance after considering transactions.

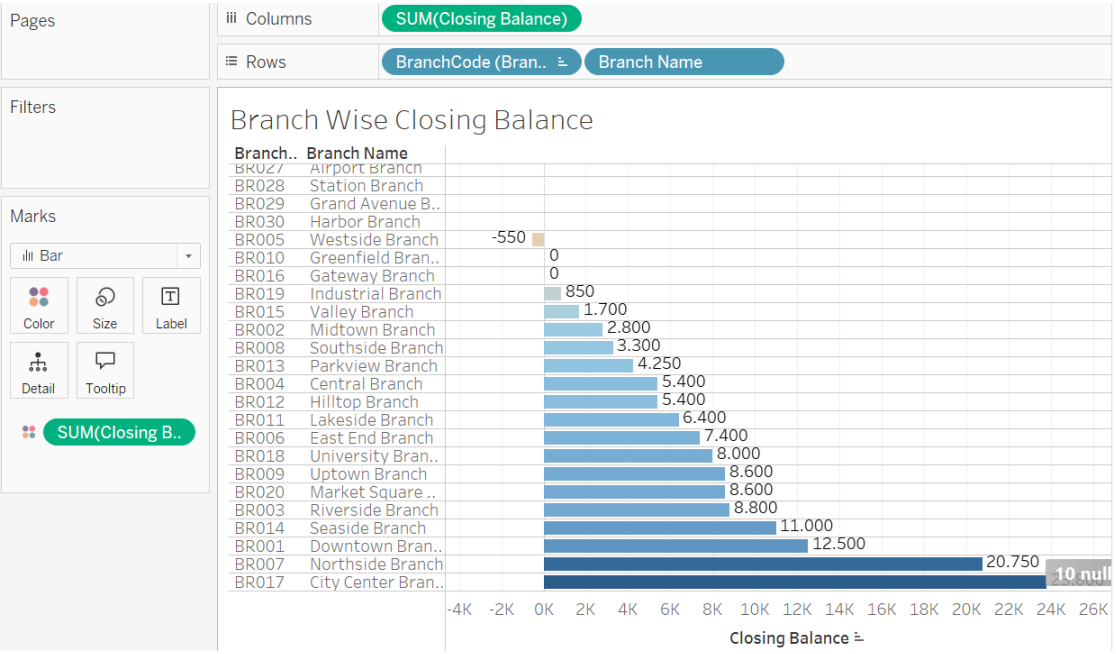
How to create report

1. New calculated measure for closing balance using formula

   ```
   [Balance] + CASE [Transaction Type]
     WHEN 'Deposit' THEN [Transaction Amount]
     ELSE -1 * [Transaction Amount]
   END
   ```

2. Rows using customer name
3. Columns using measure names
4. Measure values using SUM balance and SUM closing balance
5. Marks using text for measure values

Report specifications (4 of 6)



- This Branch Wise Closing Balance report in Tableau visualizes the Closing Balances of different branches

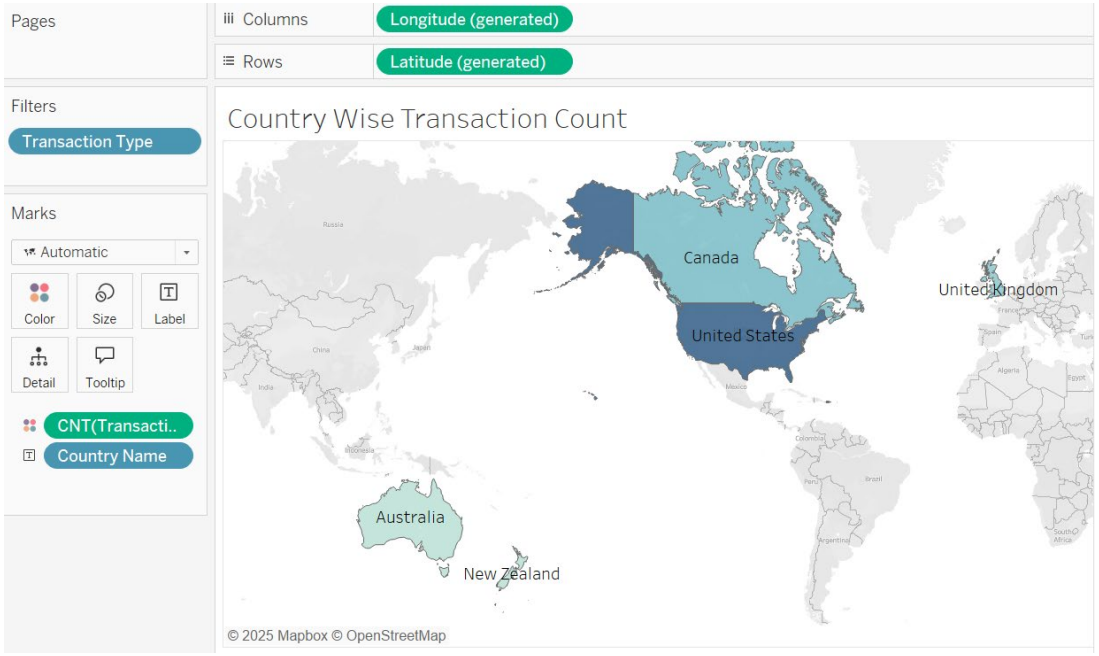
How to create report

1. New calculated measure for closing balance using formula

   ```
   [Balance] + CASE [Transaction Type]
     WHEN 'Deposit' THEN [Transaction Amount]
     ELSE -1 * [Transaction Amount]
   END
   ```

2. Rows using branch code and branch name
3. Columns using SUM of closing balance
4. Marks using color for closing values
5. Data label to make bar chart readable

Report specifications (5 of 6)

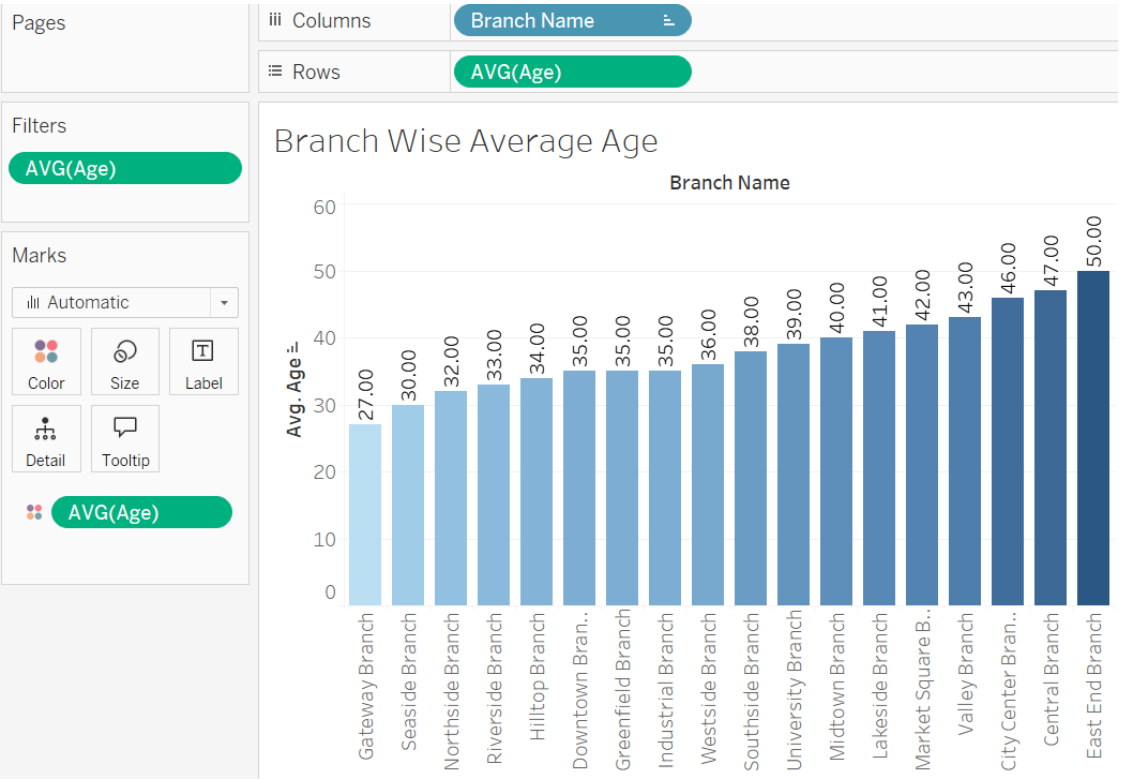


- This Tableau report visualizes the average age of customers across different branches, allowing businesses to analyze demographic distribution

How to create report

1. Columns and Rows using longitude and latitude (auto generated by tableau)
2. Coloring country area by counting transactions occurred in a country
3. Data label by country name to make map chart readable
4. Filtered by transaction type to only show country with some transactions

Report specifications (6 of 6)



- Country-Wise Transaction Count report in Tableau provides a geographical visualization of transaction volumes across different countries.

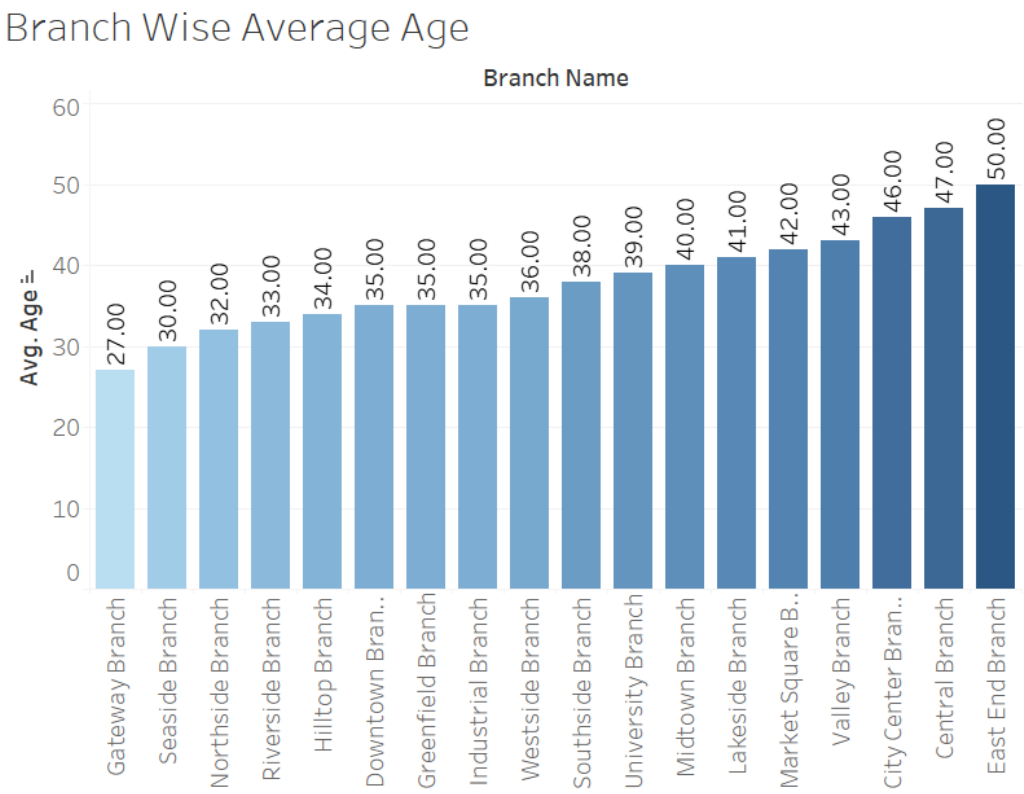
How to create report

1. New calculated measure for AGE using formula
   DATEDIFF('year', [Date Of Birth], TODAY())
2. Columns using branch name
3. Rows using average of AGE
4. Marks using color to differentiate age average
5. Filters to only shows non null values

Performance analysis guidelines (1 of 3)

- Customer wise Closing Balance
  - provides an overview of customer balances, showing both their initial Balance and the Closing Balance after considering transactions.
  - [Balance] + CASE [Transaction Type] WHEN 'Deposit' THEN [Transaction Amount] ELSE -1 * [Transaction Amount] END
- Branch Wise Closing Balance
  - visualizes the Closing Balances of different branches
  - [Balance] + CASE [Transaction Type] WHEN 'Deposit' THEN [Transaction Amount] ELSE -1 * [Transaction Amount] END
- Country Wise Transaction Count
  - provides a geographical visualization of transaction volumes across different countries.
  - Counting sum of transaction occurred in a country
- Branch wise average age
  - visualizes the average age of customers across different branches, allowing businesses to analyze demographic distribution
  - Counting average of age for each branch
- Average Transaction Value
  - Identifies spending patterns and pricing effectiveness.
  - Average of a transaction

Performance analysis guidelines (2 of 3)



Design

- Chart Type: Bar Chart
- X-Axis (Category): Branch Name
- Y-Axis (Metric): Average Age
- Color Encoding: Gradient from light blue (lower) to dark blue (higher)
- Labels: Average Age

Purpose

- To compare the average age of customers across different branches.
- Helps in demographic analysis for targeted marketing and customer service strategies.
- Assists in branch-level decision-making based on the customer base's age distribution.

Insights

- Youngest customer base: Gateway Branch (Avg Age = 27) May require youth-oriented products/services.
- Oldest customer base: East End Branch (Avg Age = 50) Likely serves older customers who may have different preferences.
- General Trend: Age increases across branches, suggesting regional or operational differences in customer demographics.
- Strategic Use:
  - Tailor marketing campaigns based on age preferences.
  - Adjust product offerings to match demographic needs.
  - Optimize customer service approaches for different age groups.

Performance analysis guidelines (3 of 3)

- **MAINTENANCE**

Refreshing Data:

1. **Automate Data Extraction** from sources (SQL, Excel, Cloud).
2. **Schedule Refresh** (daily, weekly) to keep insights up to date.
3. **Validate Data Accuracy** using consistency checks (duplicate removal, missing values).

Adding/Removing KPIs:

1. **Assess Business Needs** – Ensure relevance of new KPIs.
2. **Modify Calculated Fields** in Tableau for new formulas.
3. **Update Dashboard Layout** – Adjust visual elements to fit new KPIs.
4. **Test & Validate Changes** – Check if filters and interactivity work as expected.

Performance Optimization:

1. **Optimize Data Sources** – Use extracts instead of live connections where possible.
2. **Reduce Unnecessary Calculations** – Precompute KPIs at the database level.
3. **Limit Filters & Aggregations** – Too many dynamic filters slow down performance.
4. **Use Indexed Columns** –faster query execution.

General handover documentation

Role and key responsibilities

- **Data Owners**

Data Owners are responsible for managing the entire data lifecycle, ensuring compliance with governance policies, and addressing any quality or security issues within their datasets. They have overall accountability for data integrity and are central to decision-making regarding data access and usage.

- **Data Stewards**

Data Stewards are the custodians of data quality, responsible for ensuring that all data complies with organizational standards. They regularly monitor data accuracy and completeness, working alongside Data Owners to resolve any quality issues.

- **Compliance Officers**

Compliance Officers ensure that SecureHealth adheres to all relevant regulations, including GDPR and HIPAA. They conduct regular audits, manage data subject requests, and ensure that privacy policies are consistently applied across all processes.

- **IT Security Team**

The IT Security Team implements technical security measures such as encryption, access control, and network protection. They collaborate with other roles to ensure that all security practices align with both internal policies and external regulations.

- **Data Governance Committee**

The Data Governance Committee is responsible for overseeing the entire governance strategy. They ensure that governance policies align with the organization's goals, conduct periodic reviews, and guide Data Owners and Compliance Officers.

System architecture diagram

- Data sources/ operational systems save data into their own OLTP storage.
  - Excel, CSV., etc
  - OLTP using MySQL, PostgreSQL, etc
- Many OLTP storage will be extracted and going to Extract and load phase, then normalized data will be loaded into OLAP database.
  - ETL using talend, Pentaho, apache airflow, etc
  - OLAP using PostgreSQL, DB2, Amazon Redshift, Snowflake, etc
- OLAP databases will be used for analysis using BI tools
  - Tableau, PowerBI, Metabase, Google looker, etc

Key challenges and resolutions

- Data Cleaning Issues (Nulls, Duplicates, Formatting) - Using simple but effectively formatting tools (excel, python)
- Ensuring Data Integrity Between OLTP and OLAP - Used checksum validation and logging mechanisms during ETL.
- Integration various data sources – data extraction and phase transformation for individual sources

Insights and recommendations

Insights and recommendations

- Ensure Data Quality & Consistency
  - Implement data validation rules to check for missing, duplicate, or incorrect entries before loading data into OLAP.Standardize naming conventions across datasets (e.g., product categories, date formats, customer IDs).Perform regular audits to maintain data integrity.
- Improve Dashboard Performance & User Experience
  - Use aggregated tables or materialized views to store precomputed results
  - Convert live connections to extracts when possible to reduce query execution time.
  - Implement filters and parameters to allow users to focus on relevant data instead of loading everything at once.