University for the Creative Arts

BERLIN SCHOOL OF BUSINESS & INNOVATION

Essay / Assignment Title:

Database System Design for a South America Telecommunications Service Provider

Programme title:

MSc Data Analytics

Name:

HERNAN ENRIQUE GUZMAN MORATTO

Year:

2025

CONTENTS

Contents

Figures and Tables.

List of Figures:

List of Tables:

Statement of compliance with academic ethics and the avoidance of plagiarism

I honestly declare that this dissertation is entirely my own work and none of its part has been copied from printed or electronic sources, translated from foreign sources and reproduced from essays of other researchers or students. Wherever I have been based on ideas or other people texts I clearly declare it through the good use of references following academic ethics.

(In the case that is proved that part of the essay does not constitute an original work, but a copy of an already published essay or from another source, the student will be expelled permanently from the postgraduate program).

Name and Surname (Capital letters):

....

Date: ..../........../.........

INTRODUCTION

This study presents the design and implementation of a Data Warehouse system for “ETB”, a South American telecommunication (telecom) business with operations in 5 different countries: Colombia, Argentina, Venezuela, Chile, and Brazil. As telecom is a data-intensive industry, the ability of a well-designed data warehouse becomes essential, enabling data-driven decision-making (Bahaaudeen, 2023).

The most important aspect of any data-intense system is its schema design. Designing a scalable and logically structured data warehouse scheme tailored for a telecom company was Phase 1 of this project. The goal was to build a normalized structure that imitates business processes like customer management, usage of services, billing, support interactions, and tracking of revenue.

In Phase two the database development had place using a star-schema-driven model, the middle fact table UsageRecord was linked to key dimensions like Customer, Branch, and ServicePlan. The structure was then streamlined with Third Normal Form (3NF) mechanisms to prevent redundancy and enforce referential integrity, returning high-quality data along with optimized query performance.

In the middle of Phase 2, synthetic data was generated both manually and with Python to simulate real-world scenarios. Constraints, primary keys, and foreign keys were also enforced to support ACID properties and domain-level requirements, providing the basis for reliable analytics as well as scalability down the line.

In the final phases of the project, the data was retrieved using queries to get valuable insights out of the database, using Power BI to visualize, and to explain the results.

The GitHub link containing all related appendix is: https://github.com/HerGuzmanMoratto/ETB-Telecom-Data-Warehouse-Project

METHODOLOGY

Telecom commerce is a data-driven industry, where the importance of timely decision-making has increased. In this context, the role of a well-designed data warehouse becomes essential. Data warehouses integrate and process large amounts of data into valuable information that enlightens strategic decisions (Bahaaudeen, 2023). To accomplish this, a well-structured methodology divided into five different, but interrelated, phases was implemented to ensure strength, accuracy, and analytical capability in the database design and implementation:

Phase 1: Database Design – Schema Definition and Content Structure

This phase of the project covers the database organization by utilizing a relational schema based on normalization up to 3NF, to minimize data redundancy, improving consistency. ACID principles, Referential Integrity and Cardinality constraints were implemented to represent real-world associations accurately (Yesin et al., 2021).

As Braide and George (2025) stated, referential integrity is an important aspect of how we think about problems and how are they represented using models. Cardinality, on the other hand, is vital for accurate data representation and analytical performance (B. S. Praciano et al., 2021). The proper implementation of cardinality is crucial not only for performance, but also for reliability(Coronel and Morris, 2015).

Phase 2: Database Implementation

During the second phase of the project, the SQL schema designed in phase one is executed in MySQL Workbench version 8.0, and dummy data created using Python are populated into the tables across the five different countries and their branches to simulate real-world operations.

Constraints domains rules were employed to define business rules and ensure high quality data. Furthermore, the design sticks to the ACID principles (Atomicity, Consistency, Isolation, Durability) to ensure transactions dependability (Lotfy et al., 2016).

Phase 3: SQL Querying and Insights

Once phases one and two are completed, the database is now running and seeded. It is time to focus on extracting insights through SQL queries used to optimize retrieval time and support analytical use cases (Ahmad et al., 2019). Queries include joints, aggregations, conditionals, and nested queries to find trends in billings, customer satisfaction, seasonal spikes, payment behavior, and service usage. The queries focus on getting insights about performance by branches, customer behavior, seasonality in billing patterns, and customer support response and satisfaction.

Phase 4: Insights Visualization

This stage of the project takes the output obtained from the querying in the previous phase to transform them into visual forms such as graphics and charts using Power BI. Visualizations allow stakeholders to easily understand complex relationships and patterns within the metrics (Aung et al., 2019), and enables faster data-driven decision-making (Eberhard, 2023). Top performer branches, seasonal trends like plan usage, customer patterns such as payments behavior, customer churns, and their service satisfaction, will be presented using dashboards and visual summaries.

Phase 5: CAP Theorem Discussion

The final phase of the study dives into the analytical approach. It considers the design of the system in relation to Consistency, Availability, and Partition tolerance (CAP). Even though CAP is more applicable in distributed systems, its principles inform trade-offs in database configuration. The discussion will evaluate the way the system prioritizes Consistency and Availability over Partition tolerance based on the relational model used (Gilbert and Lynch, 2002). Ensuring efficient data retrieval through well-organized queries is crucial for strategic decision-making and operational effectiveness (Alhyasat and Al-Dalahmeh, 2013).

These five phases together enable ETB to process and analyze telecom data efficiently, driving strategic decisions through countries and branches.

PHASE 1: DATABASE DESIGN - SCHEMA DEFINITION AND CONTENT STRUCTURE.

The first phase of this project involves establishing the general schema design of the data warehouse to make sure that it follows the best practice in database modeling and the operational requirements of the telecom business. The database schema consists of 11 normalized tables, which were designed to model key entities and processes in the telecom environment. The core tables contain entities such as Customer, Branch, and Country. Additionally, tables for operational and analytical analysis were added like UsageRecord, BillingRecord, SupportTicket, RevenueTarget, ProcessLogs, ServicePlan, Employee, and Time.

The initial schema was crafted based on the star schema approach, having UsageRecord table as the main fact table at the center and linking it with appropriate dimension tables like Branch, Customer, ServicePlan, etc.

1.1. Normalization and Content Structure.

After the initial schema was defined, the best practices were followed by applying Normalization principles to reduce data redundancy and maintain atomicity within the database tables (Amato, 2023)., with all tables conforming to at least Third Normalization Form (3NF). The main goal of normalization was 3NF, making sure that every non-key record is only dependent on its primary key, aiming to minimize redundancy, enforce referential integrity, and ensure scalable data structures. The discrimination of entities such as Customer, Branch, Employee and ServicePlan demonstrates that the approach was followed, aligning with best practices in database design (Eessaar, 2016). Each table was created considering the content structure:

- Primary keys were created to establish unique identifiers.
- Relational integrity was maintained by using and properly defining foreign keys.
- ENUM and CHECK constraints were employed to validate categorical data.
- Timestamps like createdAt and updatedAt were included to enable tracking and auditability.

1.2. Referential Integrity and Cardinality.

To guarantee high-quality data, RI was enforced by establishing solid foreign key constraints to allow the relationship between records to be accurate and consistent. An instance of this is that no SupportTicket would refer to a customer that does not exist. Cardinality was defined to model actual relationships, like:

- One-to-many: A branch can have many customers.
- One-to-many: A customer can have many usage records over time.
- One-to-one: Each revenue target corresponds to a specific branch, plan, and schedule.

Defining these relationships in the database enables it to reflect real business logic in a telecom industry and optimizes the analytical queries efficiency (Dey et al., 1999).

1.3. Design Features Aligned with ACID Principles.

The database design follows the Atomicity, Consistency, Isolation and Durability principles to ensure reliable transactions:

- Atomic operations cause customer deletions or updates to execute completely or not at all.
- Consistency is applied by relationships and constraints.
- Isolation is guaranteed by transactional control on batch updates or inserts.
- Durability is supported by storing data with proper timestamps and default values.

Applying these principles together increase the system’s reliability and prevent data anomalies when used correctly (Li et al., 2022).

1.4. Domain-Specific Relevance.

Telecom organizations work with large and fast-changing datasets, which is the reason why this schema was modeled to tailor the specific needs of this business environment. It allows different specific use cases, for instance:

- Data, SMS, and Voice usage tracking per month and branch.
- Plan type and branches linked to usage trends.
- Managing support interactions and billing outcomes.

Structuring the schema this way enables the organization to derive valuable insights like service performance, employee performance, customer satisfaction and billing trends by country and branch. Figure 1 presents a graphical representation of the entity relationships, displaying the 1:1 schema designed for ETB Telecom.

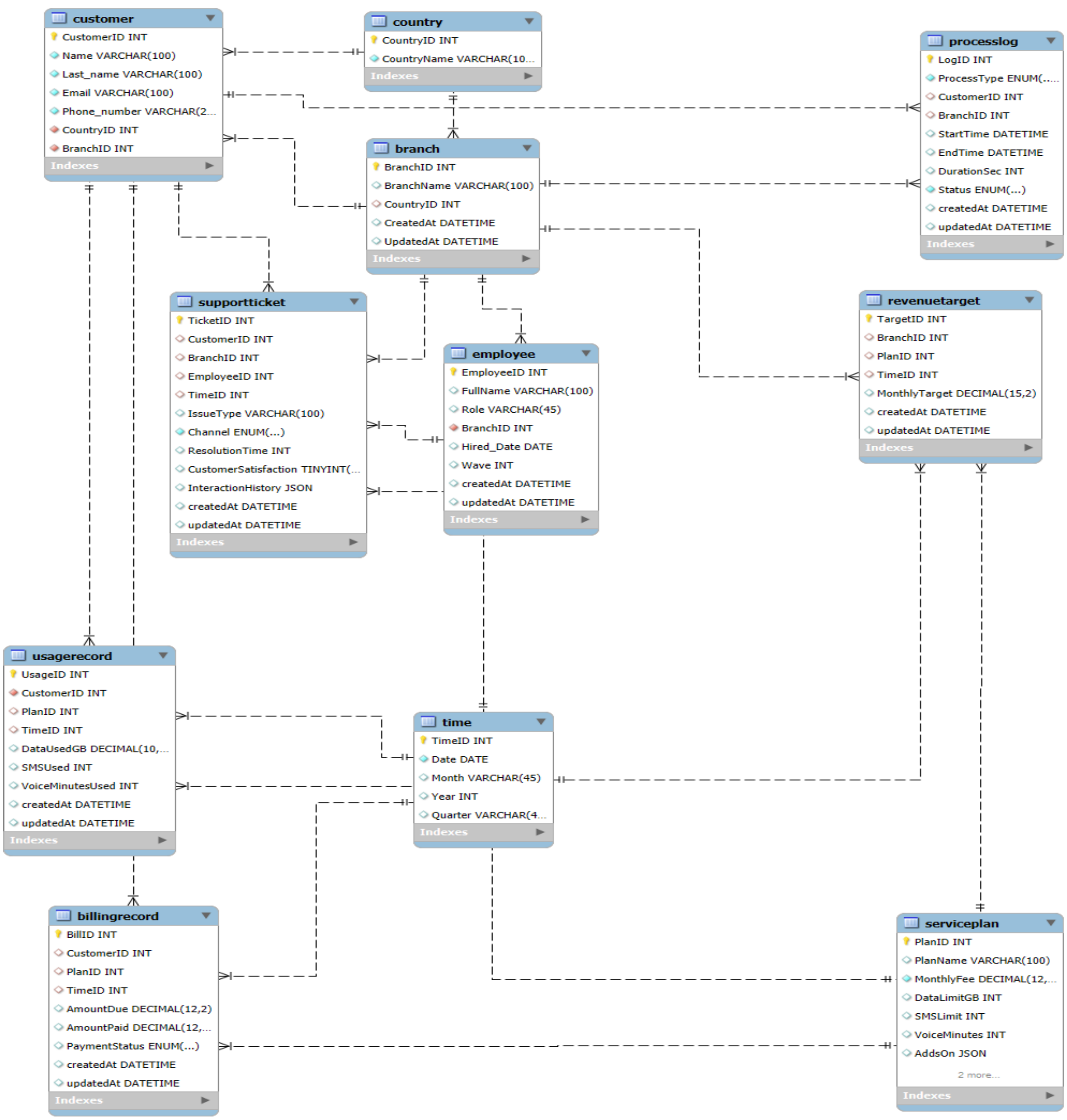

Figure 1. Entity-Relationship Diagram of the Telecom Data Warehouse Schema. Author's property.

PHASE TWO: DATABASE IMPLEMENTATION

Phase two of the project is focused on the practical implementation of the schema described in phase one, which is the creation of the tables through Data Definition Language (DDL), constrain implementation for integrity, and inserting realistic records generated manually and using python to emulate real-world scenarios in a telecom industry.

2.1. **Schema Implementation with DDL:** The schema implementation was physically done by using MySQL Workbench, following the logical design in phase 1. All tables were designed using InnoDB storage engine to support transactions and foreign key relationships, meaning that each table was crafted with:

- Primary Keys (PKs) for record identification uniquely.
- Foreign Keys (FKs) for enforcing referential integrity.
- Constraints such as NOT NULL, CHECK, and ENUM for data validation.
- Timestamps with createdAt and updatedAt columns based on CURRENT_TIMESTAMP.

2.2. **Foreign Key Strategy:** Foreign key constraints were applied based on real-world logic ensuring data consistency based on realistic business rules of a telecom company:

- ON DELETE CASCADE: Applied where child records must be deleted if the parent is deleted (e.g., branch and its employees).
- ON DELETE RESTRICT: Used to preserve historical information (e.g., TimeID in UsageRecord).
- ON DELETE SET NULL: Used when relationship is optional or association may no be longer needed (e.g., reassigning a SupportTicket when an Employee leaves.).

2.3. **Data Insertion Strategy:** As stated before, realistic synthetic data was inserted into all 11 tables, making sure to manually add the most important records such as Countries, Branches, Employes, and Customer distribution. More than 600 structured records were generated from Python-based scripts for the tables, later condensed to approximately 120 optimized entries for readability and analysis. The whole insertion DDL script is attached to the GitHub link. The main goal of the insertions was:

- Geographical coverage: Customers and branches across five countries.
- Operational roles: Each branch has a manager, tech engineer, and customer reps.
- Time-based metrics: A full year of data in the Time dimension.
- KPIs simulation: In BillingRecord, UsageRecord, and SupportTicket.

2.4. **JSON Data and Semi-Structured Use:** Tables SupportTicket and ServicePlan were expanded with JSON structured columns such as AddsOn in ServicePlan to store flexible package data and InteractionHistory in SupportTicket to simulate actual agent-customer message logs, aligning with modern telecom data strategies, allowing for structured query performance for semi-structured data. Figure 2 shows snapshots of the DDL structure where the best practices were applied, presenting the tables Customer and SupportTicket as an example.

```sql
-- ------------------------------------------------------
-- Table 3. Customer
-- ------------------------------------------------------
CREATE TABLE IF NOT EXISTS Customer (
    CustomerID INT NOT NULL AUTO_INCREMENT,
    Name VARCHAR(100) NOT NULL,
    Last_name VARCHAR(100) NOT NULL,
    Email VARCHAR(100) NOT NULL CHECK (Email LIKE '%@%.%'),
    Phone_number VARCHAR(20) NOT NULL,
    CountryID INT NOT NULL,
    BranchID INT NOT NULL,
    PRIMARY KEY (CustomerID),
    CONSTRAINT fk_customer_country FOREIGN KEY (CountryID)
        REFERENCES Country (CountryID)
        ON DELETE RESTRICT ON UPDATE NO ACTION,
    CONSTRAINT fk_customer_branch FOREIGN KEY (BranchID)
        REFERENCES Branch (BranchID)
        ON DELETE CASCADE ON UPDATE NO ACTION
) ENGINE=INNODB;
```

```sql
-- ------------------------------------------------------
-- Table 9. SupportTicket
-- ------------------------------------------------------
CREATE TABLE IF NOT EXISTS SupportTicket (
  TicketID INT NOT NULL,
  CustomerID INT,
  BranchID INT,
  EmployeeID INT,
  TimeID INT,
  IssueType VARCHAR(100),
  Channel ENUM('Phone', 'App', 'Email', 'In-Person') NOT NULL,
  ResolutionTime INT,
  CustomerSatisfaction TINYINT(1),
  InteractionHistory JSON NULL,
  createdAt DATETIME DEFAULT CURRENT_TIMESTAMP,
  updatedAt DATETIME DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
  PRIMARY KEY (TicketID),
  CONSTRAINT fk_ticket_customer
    FOREIGN KEY (CustomerID)
    REFERENCES Customer (CustomerID)
    ON DELETE CASCADE
    ON UPDATE NO ACTION,
  CONSTRAINT fk_ticket_branch
```

```sql
  CONSTRAINT fk_ticket_branch
    FOREIGN KEY (BranchID)
    REFERENCES Branch (BranchID)
    ON DELETE CASCADE
    ON UPDATE NO ACTION,
  CONSTRAINT fk_ticket_employee
    FOREIGN KEY (EmployeeID)
    REFERENCES Employee (EmployeeID)
    ON DELETE SET NULL
    ON UPDATE NO ACTION,
  CONSTRAINT fk_ticket_time
    FOREIGN KEY (TimeID)
    REFERENCES Time (TimeID)
    ON DELETE RESTRICT
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

Figure 2 DDL statements for tables Customer and SupportTicket. Author's property

2.5. **Integrity Validation and Testing:** Each table was verified through sample insertions, and foreign key integrity was verified. Additionally, all ENUM were verified, Nullability constraints and cascade rules were tested with edge cases, and all-Time dimensions and date-related inserts were aligned with TimeID. An inspection was conducted by running queries to identify orphaned records across foreign keys, ensuring proper implementation of referential integrity constraints. Figure 3 shows the queries and their output for the initial 3 relationships. All scripts used to validate referential integrity were stored alongside the DDL in the GitHub repository to ensure reproducibility.

```sql
-- 1. BillingRecord → Customer
SELECT COUNT(*) AS orphaned_customer_refs_in_billing
FROM BillingRecord b
LEFT JOIN Customer c ON b.CustomerID = c.CustomerID
WHERE c.CustomerID IS NULL;
```

orphaned_customer_refs_in_billing
0

```sql
-- 2. BillingRecord → ServicePlan
SELECT COUNT(*) AS orphaned_plan_refs_in_billing
FROM BillingRecord b
LEFT JOIN ServicePlan sp ON b.PlanID = sp.PlanID
WHERE b.PlanID IS NOT NULL AND sp.PlanID IS NULL;
```

orphaned_plan_refs_in_billing
0

```sql
-- 3. BillingRecord → Time
SELECT COUNT(*) AS orphaned_time_refs_in_billing
FROM BillingRecord b
LEFT JOIN Time t ON b.TimeID = t.TimeID
WHERE t.TimeID IS NULL;
```

orphaned_time_refs_in_billing
0

Figure 3. SQL Foreign Key Relationship Checking. Author's property.

This process was repeated for other relationships such as UsageRecord → Customer, SupportTicket → Customer, BillingRecord → ServicePlan, etc. The output of this confirmed that no orphaned references exist and that the database maintains ACID principles. Table 1 summarizes all the tables implemented, describing the information they include.

Table 1. Summary of Tables created for ETB Database. Author's property.

Table Name	Description	Table Name	Description
Country	Stores countries of operation	**Time**	Date dimension for all metrics
Branch	Business divisions per country	**UsageRecord**	Usage history (data, SMS, minutes)
Customer	All users and contact details	**BillingRecord**	Monthly charges with payment status
Employee	Staff across branches	**SupportTicket**	Customer service logs
ServicePlan	Data/Mobile plans	**RevenueTarget**	Monthly sales goals per branch
ProcessLog	Technical or service-related records		

2.6. **Transaction Control and Atomicity Testing:** Additional steps were implemented to ensure that the ETB database follows ACID principles, introducing transaction control mechanisms by:

- Managing group changes using START TRANSACTION and COMMIT.
- Undoing operations with SAVEPOINT and ROLLBACK.

- Testing Atomicity with an inducted failure nd logic validation statements.

Applying these practices is a critical step in the telecom environment, where reliability is indispensable, along with preventing anomalies, data corruption, and unauthorized changes (Mazurova et al., 2021). Figure 4 shows a screenshot of how atomicity was checked with Store Procedures transaction for a plan change for a customer that does not exist, it includes IF statements, inner queries, and CALL function to confirm if it was created correctly.

```sql
-- ================================
-- Transaction Block with Rollback and Savepoint
-- USING PRE-CHECK FOR CUSTOMER ID AND IF STATEMENTS TO COMPLETE TRANSACTIONS
-- ================================
-- ---------------------------
DROP PROCEDURE IF EXISTS safe_plan_and_billing;
DELIMITER $$
CREATE PROCEDURE safe_plan_and_billing()
BEGIN
    DECLARE customer_exists INT DEFAULT 0;
    START TRANSACTION;
    -- Update plan price
    UPDATE ServicePlan
    SET MonthlyFee = MonthlyFee + 5.00 WHERE PlanID = 2;
    SAVEPOINT sp_before_log;
    -- Check for existence of customer 150
    SELECT COUNT(*) INTO customer_exists
    FROM Customer
    WHERE CustomerID = 150 FOR UPDATE;

   -- Conditionally insert billing record
   IF customer_exists = 1 THEN
      INSERT INTO BillingRecord (BillID, CustomerID, PlanID, TimeID, AmountDue, AmountPaid, PaymentStatus)
      VALUES (999, 150, 2, 4, 25.00, 25.00, 'Paid');
   ELSE
      ROLLBACK TO sp_before_log;
   END IF;
   COMMIT;
END$$
DELIMITER ;
CALL safe_plan_and_billing();
```

Figure 4. Transaction Block with Store Procedure, Rollback and Save Point. Author's property.

2.7. **Transactional Safety and Concurrency Control:** As a final step in the database implementation process, advanced transaction management concepts were applied such as SELECT… FOR UPDATE for row-locking, SAVE POINT and ROLLBACK to prevent wrong transactions, REPETABLE READ for Isolation level adjustments, to ensure that data must stay consistent, even in high concurrency scenarios (Barthels et al., 2019).

In this scenario, a simulated transaction in which session A is inserting data into a ticket for billing issues, meanwhile session B adds a new record. In this case, session A maintains a consistent snapshot and does not reflect the committed change from session B until its transaction ends. Figure 5 displays the code used in MySQL Workbench for this scenario. Additional scenarios were tested and are included in the DDL script, part of the GitHub appendix.

```sql
-- ===================================================================
--  STEP 1: Set Isolation to REPEATABLE READ (Phantom Read Test)
SET SESSION TRANSACTION ISOLATION LEVEL REPEATABLE READ;
START TRANSACTION;

-- Count support tickets for 'Billing Issues'
SELECT
    COUNT(*) AS before_count
FROM
    SupportTicket
WHERE
    IssueType = 'Billing Issues';

-- Wait (simulate  delay)
SELECT SLEEP(10);

-- Count again (expecting no change under REPEATABLE READ)
SELECT
    COUNT(*) AS after_count
FROM
    SupportTicket
WHERE
    IssueType = 'Billing Issues';
-- Do not commit yet, session two should be run meanwhile
```

```sql
-- ===================================================================
-- STEP 2: Insert a new phantom row (SESSION B ) while SESSION A is sleeping
-- THIS SHOULD BE RUN IN A DIFFERENT QUERY TAB TO APPLY.
START TRANSACTION;

-- Insert a new ticket for the same issue type
INSERT INTO SupportTicket (
    TicketID, CustomerID, BranchID, EmployeeID, TimeID, IssueType, Channel,
    ResolutionTime, CustomerSatisfaction, InteractionHistory, createdAt, updatedAt
) VALUES (
    999, 1, 101, 2, 1, 'Billing Issues', 'App', 20, 5,
    '{"messages":[{"time":"2025-12-01T10:00:00","text":"New billing issue added."}]}',
    CURRENT_TIMESTAMP, CURRENT_TIMESTAMP
);

COMMIT;
```

Figure 5. Phantom read test snapshots. Author's property.

PHASE THREE: SQL QUERYING AND INSIGHTS.

This phase of the project focuses on extracting knowledge from the ETB Telecom Data Warehouse through SQL queries. In this context, querying helps to transform raw data into insights that allows companies to make decisions by monitoring activities, revenue monitoring on daily basis, customer segmentation, support performance evaluation, and plan optimization, enhancing customer experience by tracking customer satisfaction and answer times.

SQL plays a crucial role in performing aggregation, joining tables, and getting performance metrics. Effective querying enables knowledge discovery through statistical summarization and cross-dimensional analysis (Tanimura, 2021).

A set of 10 different queries was developed to obtain meaningful insights that can help ETB to optimize their processes and anticipate the expected customer behavior. Each query is tailored to explore a specific Key Performance Indicator (KPI) or operational aspect of the company, as summarized in Table2.

An additional bonus query was created to assess branches with above average revenue performance; it is included in the GitHub link. The queries' output will be shown in Phase Four, where they will be visualized and analyzed to help ETB to make decisions.

Table 2. Summary of queries to develop. Author's property.

Query	Title	What It Does	Tables Used or Joined
1	Total Customers per Country and Branch	Summarizes total customers by country and business unit	Customer, Branch, Country
2	Service Plan Popularity by Country and Branch	Tracks the most subscribed service plans for marketing evaluation	Customer, BillingRecord, ServicePlan, Branch, Country
3	Revenue by Month and Branch	Tracks monthly revenue trends across branches, identifies seasonal patterns	BillingRecord, Time, Customer, Branch
4	Payment Status by Country and Branch	Checks billing performance by status (Paid, Pending, Late) across countries	BillingRecord, Customer, Branch, Country
5	Average Usage per Plan by Month	Supports marketing and infrastructure decisions by analyzing usage trends	UsageRecord, Customer, BillingRecord, ServicePlan, Time
6	Revenue Target vs Actual Revenue	Evaluates if branches are meeting revenue targets	RevenueTarget, BillingRecord, Customer, Branch, Time
7	Ticket Resolution Time by Channel	Assesses which support channels resolve customer issues fastest	SupportTicket, Branch, Country
8	High-Activity Customers (≥3 Tickets)	Identify customers with high numbers of support interactions	SupportTicket, Customer, Branch
9	Top Performance in Customer Service Satisfaction	Identifies top-performing branches in service satisfaction	SupportTicket, Employee, Branch
10	Bottom Performance in Customer Service Satisfaction	Identifies lowest-performing branches to target improvements	SupportTicket, Employee, Branch
11	Bonus query: Avg Revenue	Finds branches with above average revenue performance	BillingRecord, Branch, Country, Customer

Figure 6 shows a snapshot of Query 1: Total Customer per Country and Branch and the query output. Figure 7 shows an image of Query 8: High-Activity Ticket Customers by branch, contains the DDL script and the output of it. And Figure 8 displays the DDL for Bonus Query: Branches with Above-Average Revenue Per Country and the output, comparing the branch's revenue against the average revenue of branches in the same country.

```sql
-- =====================================================
SELECT
    c.CountryName,
    b.BranchName,
    COUNT(cu.CustomerID) AS TotalCustomers,
    ROUND((COUNT(cu.CustomerID) * 100.0) / (SELECT COUNT(*)  FROM Customer), 2) AS PercentageOfTotal
FROM Customer cu
JOIN Branch b ON cu.BranchID = b.BranchID
JOIN Country c ON b.CountryID = c.CountryID
GROUP BY c.CountryName ,
 b.BranchName
ORDER BY PercentageOfTotal DESC;
```

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

CountryName	BranchName	TotalCustomers	PercentageOfTotal
Colombia	Bogotá Central	9	18.00
Brazil	São Paulo North	9	18.00
Brazil	Rio de Janeiro South	8	16.00
Colombia	Medellín Norte	6	12.00
Argentina	Buenos Aires North	5	10.00
Chile	Santiago Central	5	10.00
Venezuela	Caracas Central	4	8.00
Venezuela	Maracaibo Central	4	8.00

Figure 6 Query 1: Total Customer per Country and Branch snapshot. Author's property.

```sql
SELECT
    BranchName,
 --   cu.CustomerID,
    cu.Name,
    cu.Last_name,
    COUNT(st.TicketID) AS TicketCount
FROM SupportTicket st
JOIN Customer cu ON st.CustomerID = cu.CustomerID
JOIN Branch b ON b.BranchID = cu.BranchID
GROUP BY cu.CustomerID
HAVING TicketCount >= 3
ORDER BY TicketCount DESC;
```

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

BranchName	Name	Last_name	TicketCount
Caracas Central	Luis	Ramos	6
Santiago Central	Tomás	Navarro	5
Bogotá Central	Camilo	Jiménez	4
Buenos Aires North	Brenda	Mendoza	4
Caracas Central	María	Pérez	4
Maracaibo Central	Jorge	Torres	4

Figure 7. High-Activity Ticket Customers by branch. Author’s property.

```sql
SELECT
    b.BranchID,
    b.BranchName,
    c.CountryName,
    SUM(br.AmountPaid) AS TotalRevenue
FROM Branch b
JOIN Country c ON b.CountryID = c.CountryID
JOIN Customer cu ON cu.BranchID = b.BranchID
JOIN BillingRecord br ON br.CustomerID = cu.CustomerID
GROUP BY b.BranchID, c.CountryName
HAVING TotalRevenue > (
    SELECT AVG(sub.TotalBranchRevenue)
    FROM (
        SELECT
            b2.BranchID,
            SUM(br2.AmountPaid) AS TotalBranchRevenue
        FROM Branch b2
        JOIN Country c2 ON b2.CountryID = c2.CountryID
        JOIN Customer cu2 ON cu2.BranchID = b2.BranchID
        JOIN BillingRecord br2 ON br2.CustomerID = cu2.CustomerID
        WHERE c2.CountryName = c.CountryName
        GROUP BY b2.BranchID
    ) AS sub
)
ORDER BY c.CountryName, TotalRevenue DESC;
```

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

BranchID	BranchName	CountryName	TotalRevenue
502	Rio de Janeiro South	Brazil	308.49
102	Medellín Norte	Colombia	360.32
301	Caracas Central	Venezuela	238.00

Figure 8. Bonus Query: Branches with Above-Average Revenue Per Country. Author's property.

PHASE FOUR: INSIGHTS VISUALIZATION.

Phase four consists of data visualization to transform the organized data obtained from the analytical queries developed in previous phase into interactive business decision-making insights. Looking for a tool to build insights, Power BI appeared to be the easiest to use and is SQL-based, which is key to delivering high-quality, dynamic visual dashboards.

Design, Organization and Insights: The Power BI report contains seven thematic tabs, each one tailored to specific queries developed in Phase Three and design to respond to key business relevant questions in the telecom environment.

- **Tab 1. Customer Overview:** Built on Query 1 and 2, Figure 9 displays a map visualization highlighting the geographical location of business units. The tables outline customer distribution by branch and country, and service plan popularity by country, and region. Colombia and Brazil represent the largest percentage of the total business operation, with 15 and 17 customers out of 50, respectively. The most popular plan is Postpaid Standard with 48 subscriptions.

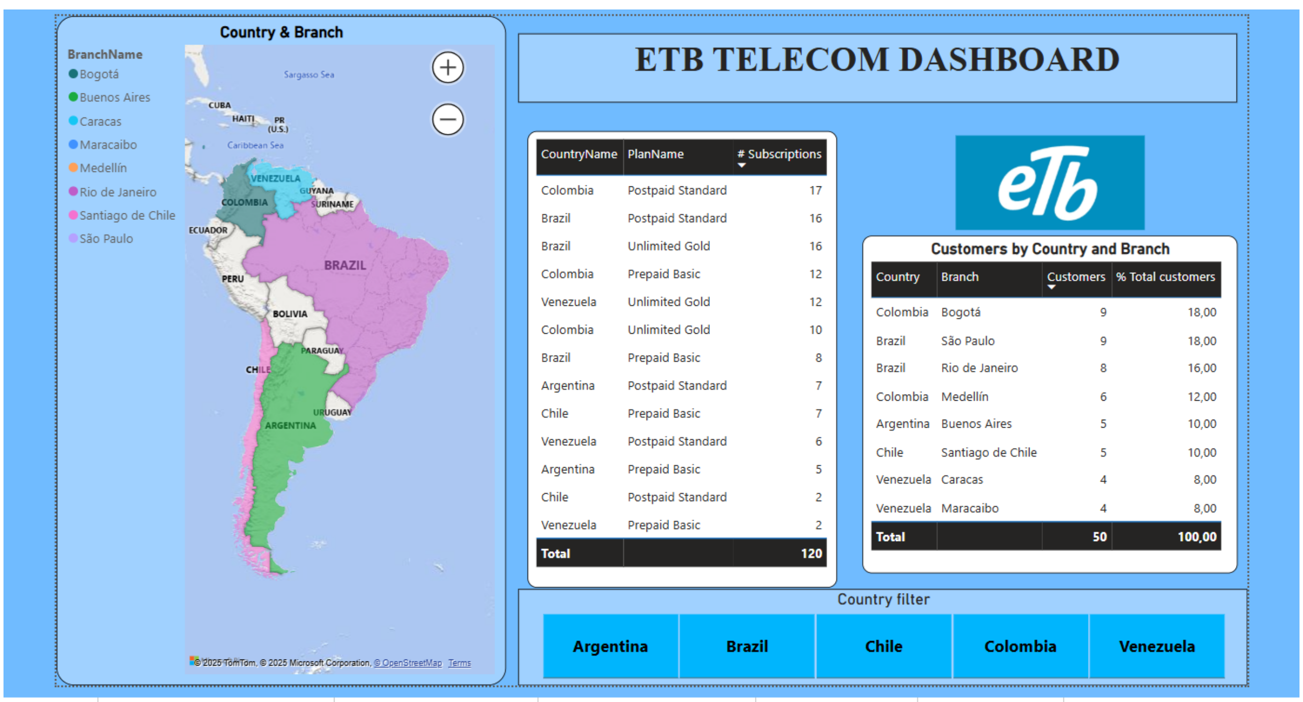

Figure 9. Customer Overview. Author's property.

- **Tab 2. Revenue by Month and Branch:** It was built from query 3, the clustered column chart shown in Figure 10 ilustrates the revenue per month by every branch for 12 months, with spikes in September for Caracas branch, and August and April for Medellin. The chart shows some lows in January, May and March.

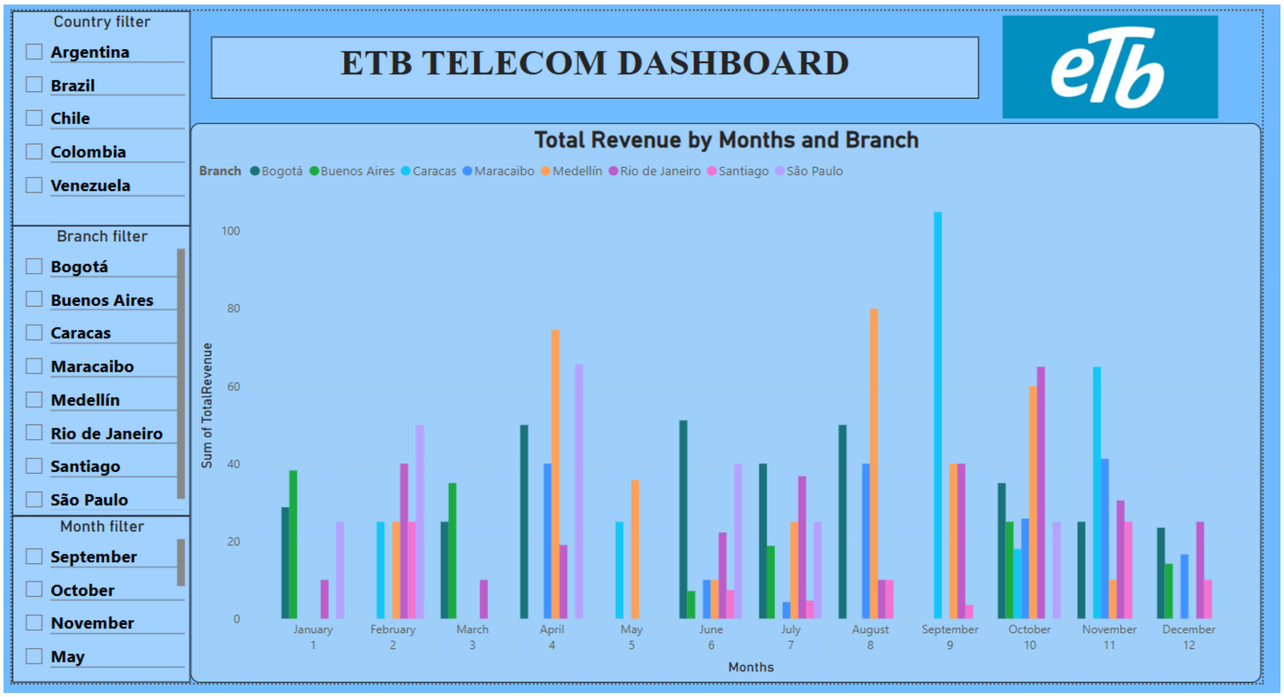

Figure 10. Revenue by Month and Branch. Author's property.

- **Tab 3. Actual vs Target Revenue Performance:** Based on Queries 6 and 11, visuals shown in Figure 11 Revenue Target Performance informs that the company is struggling to achieve monthly targets most of the time, except in months like October, November, and April. Countries above average revenue chart shows that Colombia, Brazil, and Venezuela perform well compared to other countries. Medellín and Caracas are performing well.

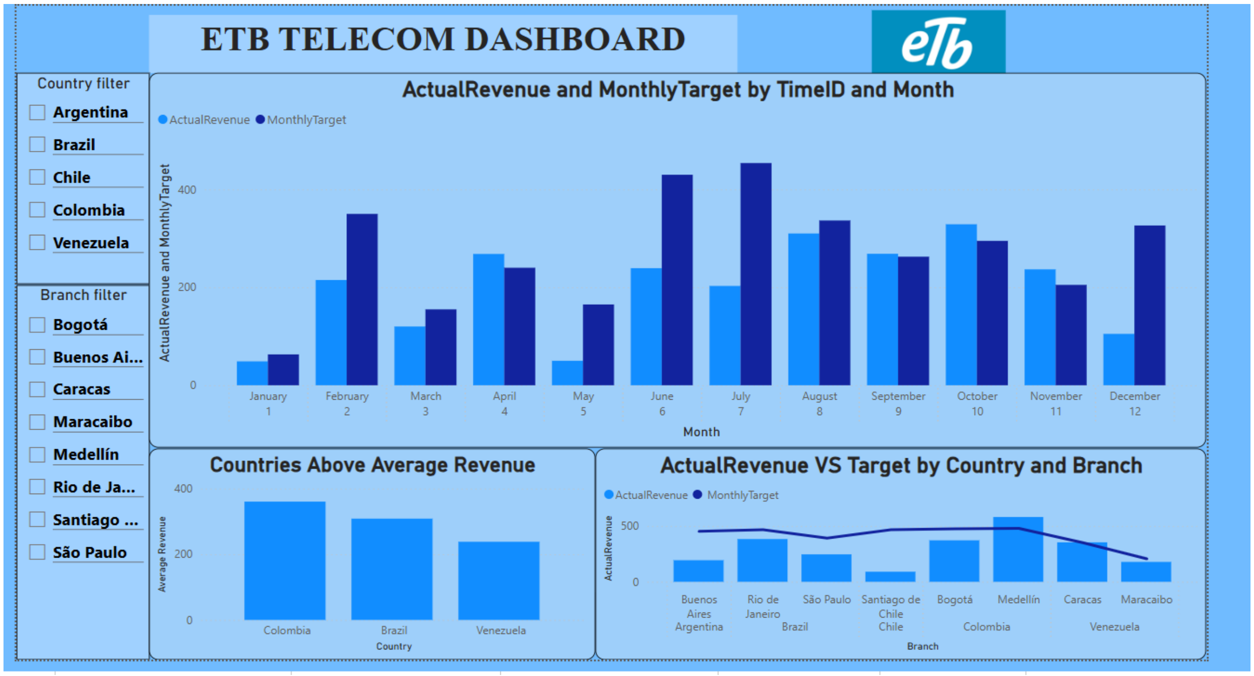

Figure 11. Revenue Target Performance. Author's property.

- **Tab 4. Payment Distribution:** Built on query 4, Figure 12 displays Tab 4, with a column chart highlighting Brazil and Colombia with the highest number of payments, but Brazil has many pending payments. The donut chart yells that only 60% of the total bill has been paid, the remaining are pending or late.

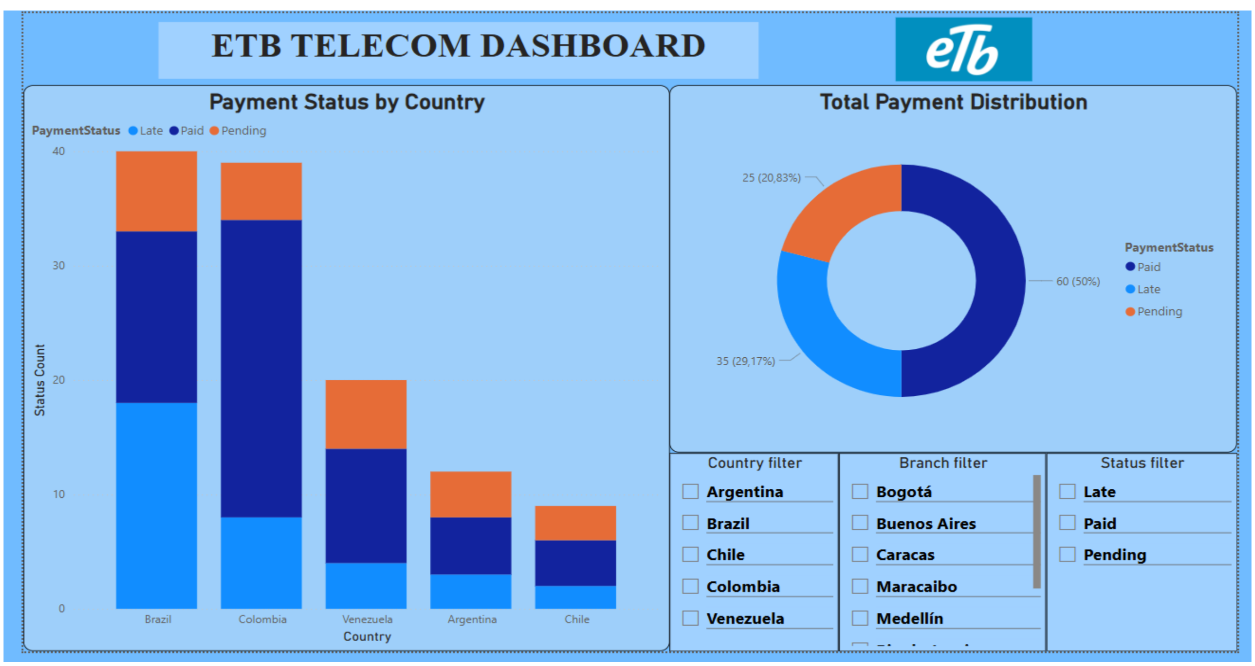

Figure 12. Payment Distribution. Author's property.

- **Tab 5. Usage Distribution:** Using Query 5, Figure 13 was built with a line chart displaying usage spikes in September and November, and unusual SMS usage in February. The chart informs that Venezuela has the highest DataGB and SMS usage.

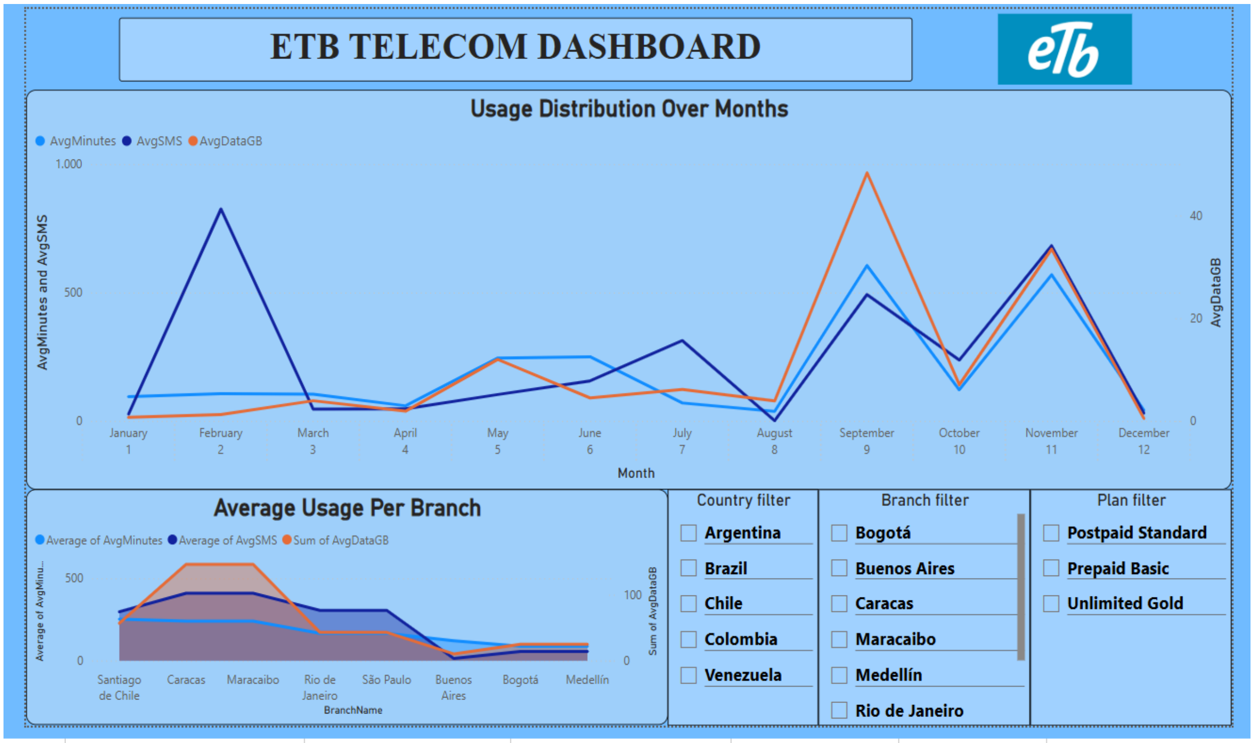

Figure 13. Usage Distribution. Author's property.

- **Tab 6. Resolution Time Analysis:** As a result of query 7, Figure 14 was created to focus on customer support efficiency. It compares average resolution times across branches with Caracas with the longest resolution time. Email support delivers the fastest resolutions overall, while app support is slower.

- **Tab 7. Customer Service Satisfaction Analysis:** based on queries 8, 9 and 10, Figure 15 shows the bar charts of top performer branches in terms of customer satisfaction. It also shows A small group of repeat callers, suggesting a need for proactive support.

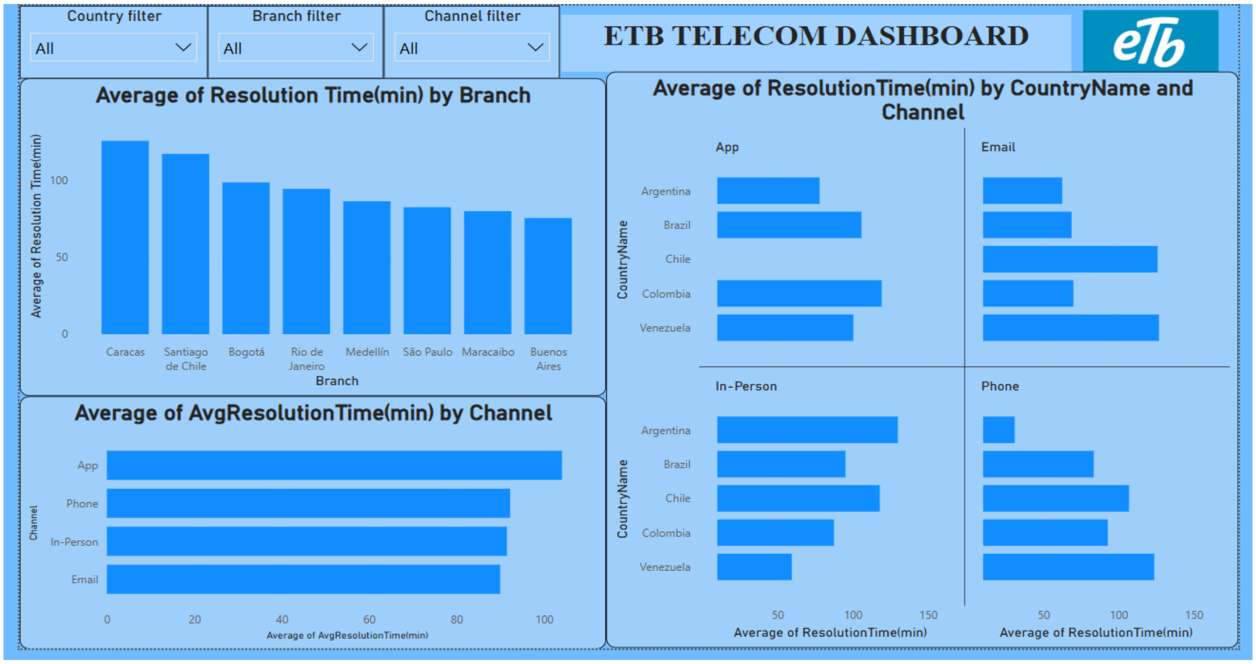

Figure 14. Resolution Time Analysis. Author's property.

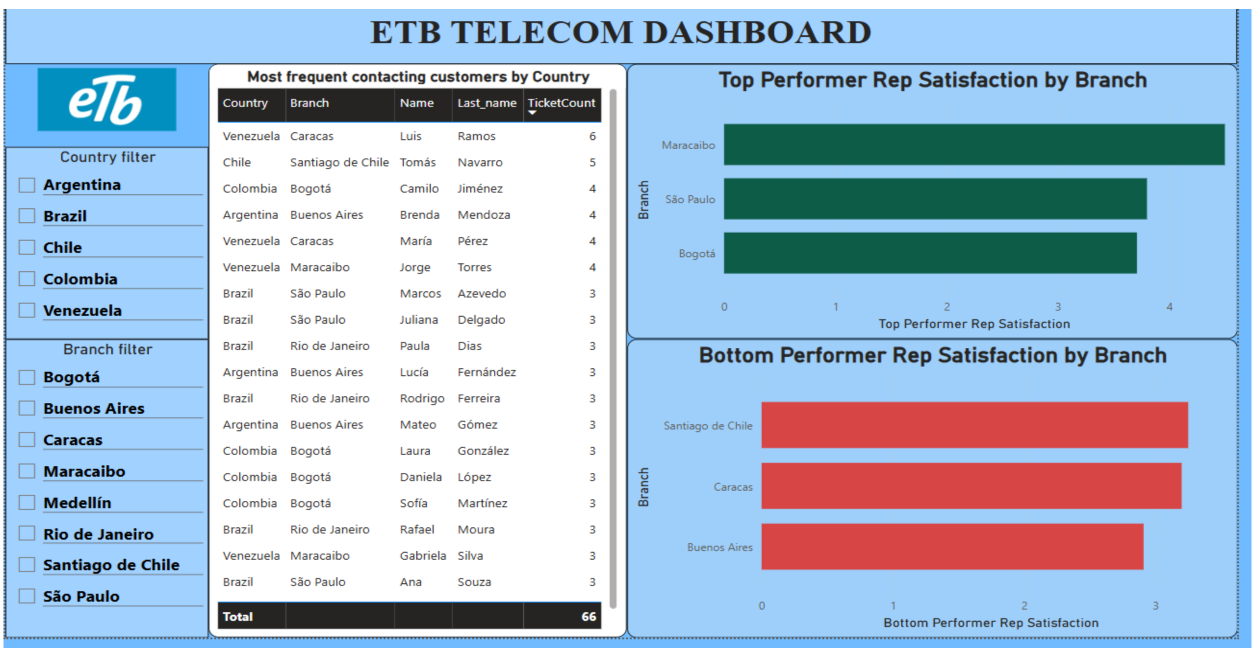

Figure 15. Customer Service Satisfaction Analysis. Author's property.

As benefits of this phase, ETB Telecom can start to develop strategies to strengthen the weaknesses detected. Visual analytics enhanced the ability to detect geographic and seasonal trends in revenue and usage, evaluating branch and plan performance quickly using visual benchmarks. The PBI file is also included in the GitHub repository for this project.

PHASE FIVE: CAP THEOREM DISCUSSION

The CAP Theorem, introduced by Brewer (2000), asserts that distributed systems can provide at most two of the following properties from the list of Consistency, Availability, and Partition Tolerance.

This study consists of a centralized MySQL database, Consistency and Availability are guaranteed through stringent transactional control, foreign key referential integrity constraints, and ACID-compliant practices. For data warehousing, focused on batch processing and historical reporting, Consistency and Availability remain optimal, with real-time partition resiliency irrelevant (Gilbert and Lynch, 2002). This database architecture is thus aligned with CAP principles under centralized design.

CONCLUDING REMARKS.

- This project contains the development of a telecom data warehouse from schema design to insight generation, from the database creation, through data input, query creation and insights visualization.
- Using normalized tables, referential integrity, and transactional controls, the backend was built with a solid foundation as per actual telecom practices.
- Analytical SQL queries were developed to extract meaningful information visualized through Power BI Dashboards.
- Visualizations revealed patterns in customer distribution, service plan popularity, revenue patterns, payment trends, support performance, and satisfaction across branches.
- By respecting CAP constraints and ACID principles, the system was standardized, offering a stable platform for informed business decisions.
- This project leaves the window open for future investigations regarding strategies to improve the weak areas when it comes to queries, insights and visualization.

BIBLIOGRAPHY

1. Ahmad, A.K., Jafar, A., Aljoumaa, K., 2019. Customer churn prediction in telecom using machine learning in big data platform. J Big Data 6, 28. https://doi.org/10.1186/s40537-019-0191-6

2. Alhyasat, E.B., Al-Dalahmeh, M., 2013. Data Warehouse Success and Strategic Oriented Business Intelligence: A Theoretical Framework. https://doi.org/10.48550/arXiv.1307.7328

3. Amato, N., 2023. Mastering database normalization: A comprehensive exploration of normal forms 14.

4. Aung, T., Niyeha, D., Heidkamp, R., 2019. Leveraging data visualization to improve the use of data for global health decision-making. J Glob Health 9, 020319. https://doi.org/10.7189/jogh.09.020319

5. B. S. Praciano, F.D., Amora, P.R.P., Abreu, I.C., Pereira, F.L.F., Machado, J.C., 2021. Robust Cardinality: a novel approach for cardinality prediction in SQL queries. J Braz Comput Soc 27, 11. https://doi.org/10.1186/s13173-021-00115-9

6. Bahaaudeen, A., 2023. The Impact of the Data Warehouse on Decision Making Quality and Speed in Higher Education, in: 2023 International Conference on IT Innovation and Knowledge Discovery (ITIKD). Presented at the 2023 International Conference on IT Innovation and Knowledge Discovery (ITIKD), IEEE, Manama, Bahrain, pp. 1–10. https://doi.org/10.1109/ITIKD56332.2023.10099501

7. Barthels, C., Müller, I., Taranov, K., Alonso, G., Hoefler, T., 2019. Strong consistency is not hard to get: two-phase locking and two-phase commit on thousands of cores. Proc. VLDB Endow. 12, 2325–2338. https://doi.org/10.14778/3358701.3358702

8. Braide, T.R., George, I.C., 2025. ENHANCING DATABASE RELIABILITY THROUGH PARTIAL REFERENTIAL INTEGRITY ENFORCEMENT. Ayden Journal of Intelligent System and Computing 13, 13–19. https://doi.org/10.5281/zenodo.15431634

9. Brewer, D.E.A., 2000. Towards Robust Towards Robust Distributed Systems Distributed Systems.

10. Coronel, C., Morris, S., 2015. Database systems: design, implementation, and management, 12e ed.

11. Dey, D., Storey, V.C., Barron, T.M., 1999. Improving database design through the analysis of relationships. ACM Trans. Database Syst. 24, 453–486. https://doi.org/10.1145/331983.331984

12. Eberhard, K., 2023. The effects of visualization on judgment and decision-making: a systematic literature review. Manag Rev Q 73, 167–214. https://doi.org/10.1007/s11301-021-00235-8

13. Eessaar, E., 2016. The Database Normalization Theory and the Theory of Normalized Systems: Finding a Common Ground 30.

14. Gilbert, S., Lynch, N., 2002. Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services. SIGACT News 33, 51–59. https://doi.org/10.1145/564585.564601

15. Li, H., Chen, Y., Li, X., 2022. Coo: Consistency Check for Transactional Databases. https://doi.org/10.48550/arXiv.2206.14602

16. Lotfy, A.E., Saleh, A.I., El-Ghareeb, H.A., Ali, H.A., 2016. A middle layer solution to support ACID properties for NoSQL databases. Journal of King Saud University - Computer and Information Sciences 28, 133–145. https://doi.org/10.1016/j.jksuci.2015.05.003

17. Mazurova, O., Naboka, A., Shirokopetleva, M., 2021. RESEARCH OF ACID TRANSACTION IMPLEMENTATION METHODS FOR DISTRIBUTED DATABASES USING REPLICATION TECHNOLOGY. ITSSI 19–31. https://doi.org/10.30837/ITSSI.2021.16.019

18. Tanimura, C., 2021. SQL for Data Analysis: Advanced Techniques for Transforming Data into Insights. O'Reilly Media, Inc.

19. Yesin, V., Karpinski, M., Yesina, M., Vilihura, V., Warwas, K., 2021. Ensuring Data Integrity in Databases with the Universal Basis of Relations. Applied Sciences 11, 8781. https://doi.org/10.3390/app11188781

APPENDIX.

1. GitHub Repository link:

https://github.com/HerGuzmanMoratto/ETB-Telecom-Data-Warehouse-Project