Employee Earnings, Absences, Performance & Retention Analysis Using SQL Server

Maksuda E Elahi
Shara Khandakar
Business Intelligence and System's Infrastructure
School of Advanced Technology
Ottawa, Ontario, Canada
elah0010@algonquinlive.com
khan0611@algonquinlive.com

Murk Asad
Anam Vakil
Business Intelligence and System's Infrastructure
School of Advanced Technology
Ottawa, Ontario, Canada
asad0022@algonquinlive.com
vaki0005@algonquinlive.com

***Abstract*— This project presents a complete SQL-based analytical workflow developed in SQL Server Management Studio (SSMS) to evaluate employee earnings, absences, departmental performance, and retention patterns using a combined dataset derived from City Employee Earnings and UCI Absenteeism at Work. All stages including data cleaning, normalization, schema development, query-driven analytics, and advanced database programming were executed exclusively through SQL. The study transforms a single flat file into a well-structured relational database using conceptual and logical design principles, enabling efficient multi-dimensional analysis through joins, window functions, aggregates, stored procedures, and triggers. The findings reveal significant trends, including strong negative relationships between absences and earnings, seasonal patterns of absenteeism, department-level performance variations, and retention issues reflected in termination data. This work demonstrates how SQL-driven database analytics can convert raw organizational data into actionable insights that support workforce management, operational planning, and long-term organizational decision-making .**

***Index Terms*—Database Design, Employee Analytics, Earnings and Absenteeism, Retention Analysis, SQL Server**

I. INTRODUCTION

This project involves performing comprehensive database analytics on the City Employee Earnings[1] and Absences dataset [2] using SQL Server. The objective is to clean, normalize, model, and analyze the data to discover insights related to employee absences, earnings performance, departmental effectiveness, and retention patterns. The project includes data cleaning, normalization, schema design, SQL-based analytics, and insight extraction regarding employee and departmental behavior.

II. DATASET OVERVIEW

The dataset includes employee identifiers, job and department assignments, multiple components of quarterly earnings, absence records with seasonal and calendar attributes, and termination details. The dataset contains more than 6000 records and 28 columns representing earnings and absence events across several years. This dataset concatenated using two data sources. One is obtained from City Employee Earnings [1] from City of Philadelphia, United States and UCI Absenteeism at work dataset [2]. Although the dataset does not belong to a single source, this work is done for demonstration purposes, as most of the employee data is confidential, hence this data represents any kind of similar scenario in each organization.

III. DATA CLEANING

The data cleaning process involved removing empty columns, renaming attributes, converting earnings fields to numeric formats, standardizing department and job names, validating date fields, verifying foreign key consistency, and exporting the cleaned dataset for SQL Server import as shown in Figure 1 and Figure2 shows the flat file one table schema before normalization.

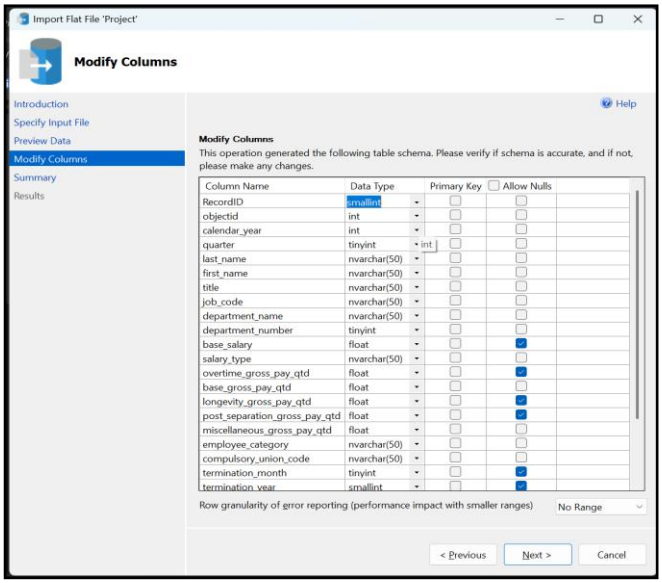

Fig. 1. Importing Data as flat file to SSMS

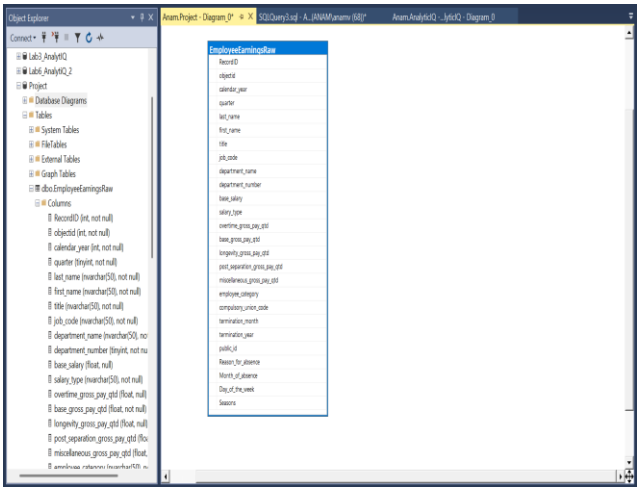

Fig. 2. Flat file before normalization

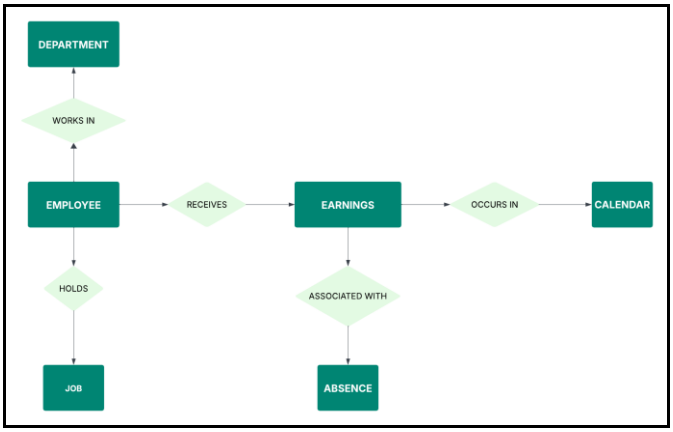

Fig. 3. Conceptual Framework

IV. CONCEPTUAL DESIGN

The conceptual design defines the high-level structure of the database before physical implementation. Since the dataset originally existed as a single flat file containing employee identifiers, job details, departmental information, earnings metrics, and absence records, a conceptual model was created to separate entities and represent how they relate within an organization. This model serves as the foundation for normalization and relational schema development.

At the conceptual level, the organization is represented by five core entities: Employee, Department, Job, Earnings, Absence, and Calendar. Each entity captures unique aspects of business information. Employees work within departments and hold specific job roles. Employees generate earnings over time, and these earnings may be associated with absences. The calendar entity provides a temporal structure that supports seasonal, monthly, and quarterly analysis.

These entities interact through well-defined relationships. An employee works in a department and holds a job. An employee earns compensation, which is stored in the Earnings entity as a fact event. Each earnings event is recorded in a specific calendar period, and may be associated with absence, enabling analysis of how attendance influences compensation. The conceptual diagram created using SQL-based reasoning reflects these relationships: Employee ↔ Department, Employee ↔ Job, Employee ↔ Earnings, Earnings ↔ Calendar, and Earnings ↔ Absence.

This conceptual model provides a clear, abstract view of the organizational data before technical details such as keys, constraints, and column definitions are introduced. It guides the logical and physical database design phases by ensuring that each entity represents a single, well-defined concept and that all relationships mirror real-world business operations. Ultimately, this conceptual framework as shown in Figure[3] was developed using lucid chart [3], and it enables a normalized, analyzable, and scalable database structure implemented entirely using SQL within SSMS.

V. LOGICAL DESIGN

A star-schema-style relational model was developed for efficient analytics. The Fact table named Earnings contains measurable data such as multiple earnings components and AbsenceID, and connects to Employee, Job, Department, and Calendar dimension tables. The detailed ERD is shown in Figure 3.

A. Normalization

Normalization was performed to achieve First, Second, and Third Normal Forms. Values were made atomic, repeating groups were eliminated, and each row represented a single event. Job, Department, and Calendar attributes were separated into their own tables to remove partial dependencies. Transitive dependencies were eliminated so that each non-key attribute depended solely on its primary key. Following is the detailed description of each normalization step.

- **First Normal Form (1NF)**
  - Ensured atomic values
  - Removed repeating groups
  - Each row represents a single employee event (earnings/absence)
- **Second Normal Form (2NF)**
  - Removed partial dependencies
  - Extracted Job, Department, and Calendar into separate tables
  - Ensuring employee attributes does not depend on fact-level attributes
- **Third Normal Form (3NF)**
  - Eliminated transitive dependencies:
  - DepartmentName depends on DepartmentID, not Employee
  - JobTitle depends on JobID
  - Calendar attributes depend only on CalendarID
  - This resulted in clean dimension tables

B. Entity Relationship Diagram

The ERD illustrates relationships between employees, departments, jobs, and earnings events. Employee tables link department and job tables, while Fact table Earnings links to dimension tables, enabling multi-dimensional SQL analysis.

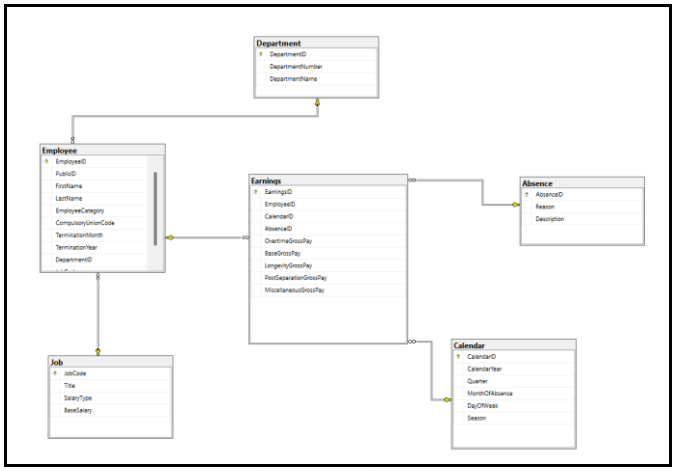

Fig. 4. ERD Diagram

C. List of Tables

- **Employee**
  Contains descriptive employee details
  - PublicID (PK)
  - FirstName
  - LastName
  - EmployeeCategory
  - CompulsoryUnionCode
  - TerminationMonth,
  - TerminationYear
  - JobCode(FK)
  - DepartmentID (FK)
- **Job**
  Stores job-related information
  - JobCode(PK)
  - JobTitle
  - SalaryType
  - BaseSalary
- **Department**
  Stores department-level attributes
  - DepartmentID(PK)
  - DepartmentName
  - DepartmentNumber
- **Calendar**
  Captures all date-related attributes
  - CalendarID(PK)
  - CalendarYear
  - Quarter
  - Month
  - DayOfWeek
  - Season
- **Absence**
  Provides details of employee's attendance
  - AbsenceID (PK)
  - Reason
  - Description
- Earnings (Main Fact Table)
  Captures details of employee earnings
  - EarningsID (PK)
  - EmployeeID (FK)
  - CalendarID (FK)
  - AbsenceID (FK)
  - OvertimeGrossPay
  - BaseGrossPay
  - LongevityGrossPay
  - PostSeparationGrossPay
  - MiscellaneousGrossPay

D. Constraints Implementation

UNIQUE constraints were applied to prevent duplicate job records. CHECK constraints enforced valid ranges for quarters, months, and salary values. DEFAULT constraints populated missing values automatically for fields such as EmployeeCategory and BaseSalary, ensuring consistent data entry.

a) UNIQUE

The UNIQUE constraint ensures that certain values cannot repeat. It blocks duplicate entries

- Applies UNIQUE for JobCode and Title, this guarantees that the same job is not accidentally added twice
- Job titles can repeat (example: job title (Police officer 1), but each job code must be tied to only
- one title, this prevents accidental duplication of job records
- Keeps the Job table clean and consistent

b) CHECK CONSTRAINTS

- CHECK prevents invalid data and ensures that the values entered into a column follow valid rules
- Quarter must be 1–4 (because a year has 4 quarters).
- Termination Month must be 1–12 or NULL
- Salary values must not be negative; it prevents impossible dates (like Month = 15 or Quarter = 7).
- Stops negative payroll values, which would break analytics.

c) DEFAULT CONSTRAINTS

- If a user does not provide a value, SQL Server automatically fills in a default value
- Applied CHECK for EmployeeCategory if data is missing (by default = 'Unknown') and if BaseSalary data is missing (by default = 0)
- Prevents empty or NULL values in important columns

- Ensures consistent data entry

VI. PHYSICAL DATABASE DESIGN

Using SQL Server Management Studio (SSMS), the ERD is implemented into physical database system using SQL as programming language. Figure 5 shows some of the tables being created in SSMS. Furthermore, analytical insights were derived entirely through SQL queries using GROUP BY, JOIN operations, aggregate functions, CASE expressions, and COALESCE for handling null values. No BI tools were used; all analysis was SQL-driven.

```sql
----Create Dimension Tables (SQL Server)

-- 1. Department Dimension
CREATE TABLE Department (
    DepartmentID INT IDENTITY(1,1) PRIMARY KEY,
    DepartmentNumber INT,
    DepartmentName NVARCHAR(100)
);

-- 2. Job Dimension
CREATE TABLE Job (
    JobCode NVARCHAR(50) PRIMARY KEY,
    Title NVARCHAR(150),
    SalaryType NVARCHAR(50),
    BaseSalary FLOAT
);

-- 3. Calendar Dimension
CREATE TABLE Calendar (
    CalendarID INT IDENTITY(1,1) PRIMARY KEY,
    CalendarYear INT,
    Quarter INT,
    MonthOfAbsence TINYINT,
    DayOfWeek NVARCHAR(20),
    Season NVARCHAR(20)
);

-- 4. Absence Dimension
CREATE TABLE Absence (
    AbsenceID INT IDENTITY(1,1) PRIMARY KEY,
    Reason NVARCHAR(200)
);
```

```sql
-- 5. Employee Dimension
CREATE TABLE Employee (
    EmployeeID INT IDENTITY(1,1) PRIMARY KEY,
    PublicID INT,
    FirstName NVARCHAR(50),
    LastName NVARCHAR(50),
    EmployeeCategory NVARCHAR(50),
    CompulsoryUnionCode NVARCHAR(50),
    TerminationMonth TINYINT,
    TerminationYear INT,
    DepartmentID INT,
    JobCode NVARCHAR(50),
    FOREIGN KEY (DepartmentID) REFERENCES Department(DepartmentID),
    FOREIGN KEY (JobCode) REFERENCES Job(JobCode)
);
```

Fig. 5. Creation of Dimension Tables in SQL Server

A. QUERIES

- **Highest Absences**

  SQL queries identified Eric Howard recorded the most absences with 3528, and the second highest number of absences was recorded for Cynthia Windfelder with 3200, while Rasheda Madison had the lowest among the top ten with 1800.Overall, This ranking highlights a wide gap between the highest- and lowest-absence employees within the top ten.

```sql
---Employees with highest absences by using public id
SELECT TOP 10
    emp.PublicID,
    emp.FirstName,
    emp.LastName,
    COUNT(e.AbsenceID) AS AbsenceCount
FROM Earnings e
JOIN Employee emp ON e.EmployeeID = emp.EmployeeID
WHERE e.AbsenceID IS NOT NULL
GROUP BY emp.PublicID, emp.FirstName, emp.LastName
ORDER BY AbsenceCount DESC;
```

77 %

Results | Messages

	PublicID	FirstName	LastName	AbsenceCount
1	7088	Eric	Howard	3528
2	10360	Cynthia	Windfelder	3200
3	13089	Pamela	Baker	2888
4	3258	Khalil	Goldsmith	2048
5	18702	Ronald	Harley	2048
6	1231	Zackery	Henderson	2048
7	736	Martha	Watkins	2048
8	37106	James	Johnson, Jr.	1936
9	9415	Christopher	Whitaker	1800
10	28626	Rasheda	Madison	1800

Fig. 6. SQL query showing highest-absence employees

- **Reason Behind Highest Absences**

  Seasonal analysis revealed that absence patterns align with predictable seasonal behavior rather than random variation. For example, Eric Howard was mostly absent during summer, while others showed higher absences in spring or fall.

```sql
---WHY these employees have high absences?
--without department name
SELECT
    emp.PublicID,
    emp.FirstName,
    emp.LastName,
    c.Season,
    COUNT(e.AbsenceID) AS AbsenceCount
FROM Earnings e
JOIN Employee emp ON e.EmployeeID = emp.EmployeeID
JOIN Calendar c ON e.CalendarID = c.CalendarID
WHERE e.AbsenceID IS NOT NULL
  AND emp.PublicID IN (7088, 10360, 13089, 18702)  -- top 4 employees
GROUP BY
    emp.PublicID, emp.FirstName, emp.LastName, c.Season
ORDER BY
    emp.PublicID, AbsenceCount DESC;
```

77 %

Results | Messages

	PublicID	FirstName	LastName	Season	AbsenceCount
1	7088	Eric	Howard	2	1512
2	7088	Eric	Howard	4	1176
3	7088	Eric	Howard	3	504
4	7088	Eric	Howard	1	336

Fig. 7. SQL output showing seasonal absence patterns

- **Absence vs. Earnings**

  Analyses showed a strong negative relationship between absences and earnings. Employees with high absences frequently had zero recorded earnings for those periods. Even moderate absences resulted in reduced earnings, illustrating how attendance directly affects productivity.

```sql
SELECT
    emp.PublicID,
    emp.FirstName,
    emp.LastName,

    -- Total absences
    SUM(
        CASE
            WHEN e.AbsenceID IS NOT NULL THEN 1
            ELSE 0
        END
    ) AS TotalAbsences,

    -- Total earnings
    SUM(
        COALESCE(e.BaseGrossPay, 0) +
        COALESCE(e.OvertimeGrossPay, 0) +
        COALESCE(e.LongevityGrossPay, 0) +
        COALESCE(e.PostSeparationGrossPay, 0) +
        COALESCE(e.MiscellaneousGrossPay, 0)
    ) AS TotalEarnings
FROM Earnings e
JOIN Employee emp ON e.EmployeeID = emp.EmployeeID
GROUP BY emp.PublicID, emp.FirstName, emp.LastName
ORDER BY TotalAbsences DESC;
```

Results | Messages

	PublicID	FirstName	LastName	TotalAbsences	TotalEarnings
1	7088	Eric	Howard	3528	0
2	10360	Cynthia	Windfelder	3200	0
3	13089	Pamela	Baker	2888	0
4	1231	Zackery	Henderson	2048	0
5	18702	Ronald	Harley	2048	0
6	736	Martha	Watkins	2048	0
7	3258	Khalil	Goldsmith	2048	0
8	37106	James	Johnson,...	1936	0
9	28626	Rasheda	Madison	1800	30000
10	9415	Christop...	Whitaker	1800	0
11	17420	Willie	Brown	1568	53200
12	9893	Gregory	Boulware	1568	84372.9619...
13	18313	James	Williams	1568	0
14	9444	Frank	Gilbert	1352	92724.3201...
15	8236	Lamont	Jeffreys	1352	50676.0788...
16	9705	Richard	Marshall	1352	90259.5205...
17	23895	Marlene	Eggaer	1352	130936

Fig. 8. Employee absence totals compared with earnings

- **Department Performance**

  Department-level earnings analysis revealed significant differences across units. PPD Police generated the highest total earnings, followed by STS Streets and the Judicial District. Departments such as Water, Parks and Recreation, and Health showed comparatively lower earnings. These variations reflect differences in workforce size, operational demands, and compensation structures across municipal departments.

```sql
SELECT
    d.DepartmentName,
    SUM(
        COALESCE(e.BaseGrossPay, 0) +
        COALESCE(e.OvertimeGrossPay, 0) +
        COALESCE(e.LongevityGrossPay, 0) +
        COALESCE(e.PostSeparationGrossPay, 0) +
        COALESCE(e.MiscellaneousGrossPay, 0)
    ) AS TotalDeptEarnings
FROM Earnings e
JOIN Employee emp ON e.EmployeeID = emp.EmployeeID
JOIN Department d ON emp.DepartmentID = d.DepartmentID
GROUP BY d.DepartmentName
ORDER BY TotalDeptEarnings DESC;
```

77 %

Results | Messages

	DepartmentName	TotalDeptEarnings
1	PPD Police	12525615.699091
2	STS Streets	7408369.91687822
3	FJD 1st Judicial District PA	4515871.91882896
4	COM Commerce - Division of Aviation	2964494.51132584
5	COM Commerce	2964494.51132584
6	PWD Water	2310800.29919243
7	FLP Free Library of Phila	1835105.3634901
8	PPR Parks and Recreation	1628319.29481316
9	MDO Managing Director Office	1491465.60643768
10	DPH Health	1109740.63419127

Fig. 9. Total earnings by department

- **Retention By Department**

  Retention was assessed using termination year attributes. Some departments displayed strong stability, while others experienced high turnover. Retention insights help identify departments that may require HR intervention or additional support.

```sql
--Retention Rate for Each Department
SELECT
    d.DepartmentName,
    SUM(CASE WHEN e.TerminationYear IS NULL THEN 1 ELSE 0 END) AS Retained,
    SUM(CASE WHEN e.TerminationYear IS NOT NULL THEN 1 ELSE 0 END) AS NotRetained,
    COUNT(*) AS TotalEmployees,
    CAST(SUM(CASE WHEN e.TerminationYear IS NULL THEN 1 ELSE 0 END) * 100.0
        / COUNT(*) AS DECIMAL(5,2)) AS RetentionRate
FROM Employee e
JOIN Department d ON e.DepartmentID = d.DepartmentID
GROUP BY d.DepartmentName
ORDER BY RetentionRate DESC;
```

77 %

Results | Messages

	DepartmentName	Retained	NotRetained	TotalEmployees	RetentionRate
1	BRT Board of Revision of Taxes	2	0	2	100.00
2	CHR Phila Comsn on Hum Relatns	4	0	4	100.00
3	LRB L and I Review Board	2	0	2	100.00
4	OLR Office of Labor	2	0	2	100.00
5	PHL Dept of Aviation	190	0	190	100.00
6	SAN Sanitation	2	0	2	100.00
7	STS Streets	4298	554	4852	88.58
8	REC Records	14	2	16	87.50
9	OFM Fleet Management	98	20	118	83.05
10	DFS Dept of Fleet Services	98	20	118	83.05

Fig. 10. Retention rate by department

- **Acid Properties**
  Table Absence consisted of incomplete information, as the reason column indicated only codes but not actual description. The description behind every reason of absence is provided in UCLA[2] website and was imported into this table by creating a new column named description. In order to save all the information permanently, a COMMIT function was performed to keep the data intact.

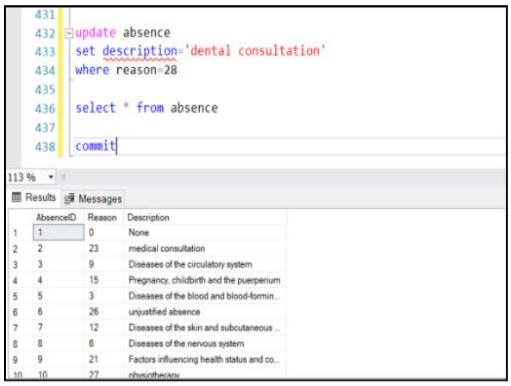

Fig. 11. SQL update and commit applied to the Absence table

VII. ADVANCED ANALYTICS

Advanced Analytics include triggers, developing stored procedures and creating indexes for fast data retrieval.

- **Triggers**
  A trigger was implemented to automatically apply payroll penalties. When a new earnings record is inserted and the associated absence reason is unapproved, MiscellaneousGrossPay is automatically reduced by $50. (Note that the unapproved absence category is a sample to test such situations in future, the dataset itself does not provide unapproved reasoning of absences). Approved absences such as vacation or medical leave are exempt from penalties. Examples are also given in Figures 12 and 13 to display how the trigger works.

```
--Change reason code
ALTER TRIGGER trg_LatePenalty
ON Earnings
AFTER INSERT
AS
BEGIN
    UPDATE e
    SET e.MiscellaneousGrossPay =
        COALESCE(e.MiscellaneousGrossPay, 0) - 50
    FROM Earnings e
    JOIN inserted i ON e.EarningsID = i.EarningsID
    JOIN Absence a ON i.AbsenceID = a.AbsenceID
    WHERE a.Reason IN (0, 26);   --selected reason code to apply penalty
END;
GO
```

Fig. 12. Trigger for payroll penalties

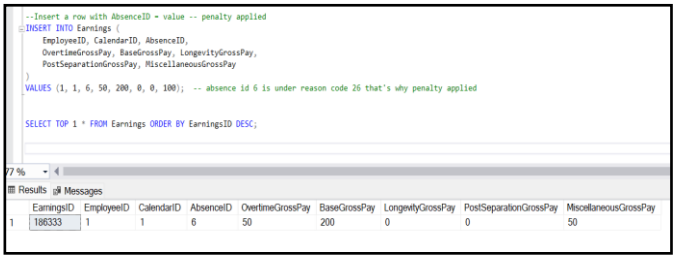

Fig. 13. Trigger execution showing automatic payroll deduction

- **Stored Procedure For Payroll Calculation**
  A stored procedure was created to calculate quarterly salaries using CalendarYear, Quarter, multiple earnings components, and COALESCE for null handling. The procedure outputs EmployeeID, FirstName, LastName, and QuarterlySalary, demonstrating parameterized SQL and modular payroll calculation logic.

```
-- Returns payroll for a given YEAR and QUARTER
CREATE PROCEDURE sp_Employee_Payroll
    @Year INT,
    @Quarter INT
AS
BEGIN
    SELECT
        emp.EmployeeID,
        emp.FirstName,
        emp.LastName,
        SUM(
            COALESCE(e.BaseGrossPay,0) +
            COALESCE(e.OvertimeGrossPay,0) +
            COALESCE(e.LongevityGrossPay,0) +
            COALESCE(e.PostSeparationGrossPay,0) +
            COALESCE(e.MiscellaneousGrossPay,0)
        ) AS QuarterlySalary
    FROM Earnings e
    JOIN Employee emp ON e.EmployeeID = emp.EmployeeID
    JOIN Calendar c ON e.CalendarID = c.CalendarID
    WHERE c.CalendarYear = @Year
      AND c.Quarter = @Quarter
      AND e.AbsenceID IS NULL    -- Only earning rows
    GROUP BY emp.EmployeeID, emp.FirstName, emp.LastName;
END;
GO
```

```
Messages
Commands completed successfully.
```

```
SELECT CalendarYear, Quarter, COUNT(*)
FROM Calendar
GROUP BY CalendarYear, Quarter
ORDER BY CalendarYear, Quarter;
```

82 % Results Messages

	CalendarYear	Quarter	(No column name)
1	2019	2	114
2	2019	3	124
3	2019	4	138
4	2020	1	128
5	2020	2	130
6	2020	3	156
7	2020	4	144

```
--execution
EXEC sp_Employee_Payroll @Year = 2019, @Quarter = 2;
```

77 % Results Messages

	EmployeeID	FirstName	LastName	QuarterlySalary
1	1	Monte	Guess	685

Fig. 14. Stored procedure creation and sample execution result

- **Indexing (Non-Clustered)**
Creating a non-clustered index on EmployeeID greatly improves query performance because it allows SQL Server to perform an index seek instead of scanning the entire Earnings table. Before indexing, the query required 1535 logical reads, meaning the engine had to read almost the whole table just to find four matching rows as shown in figure 15. After adding the index, logical reads dropped to 14, showing that SQL Server can now jump directly to the needed records. This major reduction in I/O proves that EmployeeID is a highly selective and frequently accessed column, making it an ideal choice for indexing. As a result, lookups based on employees become much faster and more efficient, especially in analytical workloads.

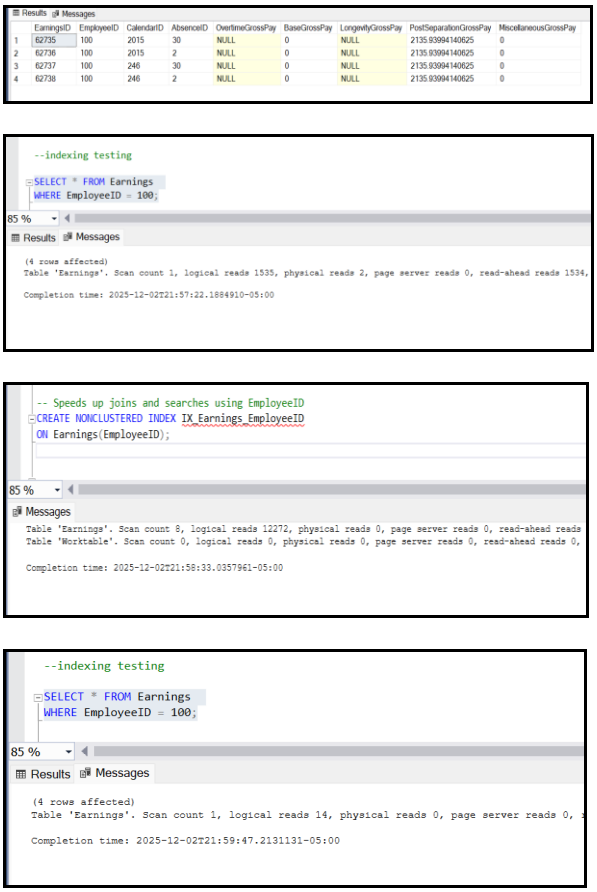

Fig. 15. Before and After Non-Clustered Index on EmployeeID

- **Window Function (Dense_Rank)**
Using DENSE_RANK, employees per department were ranked based on their absences, to evaluate their performance throughout.

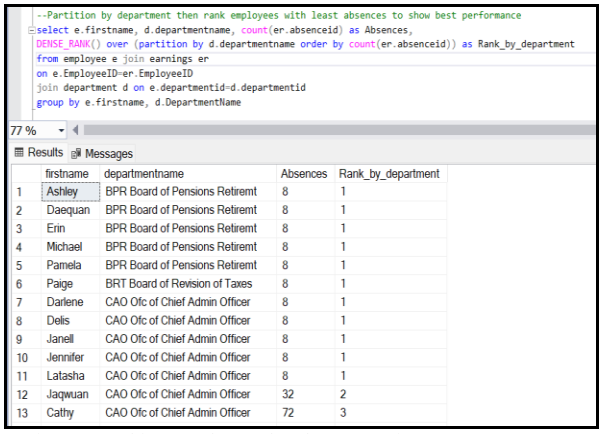

Fig. 16. Ranking employees by absences using DENSE_RANK

- **Common Table Expressions With Subquery**
Employees who were never absent, obtaining their information along with department were conducted using CTE and a subquery to support it.

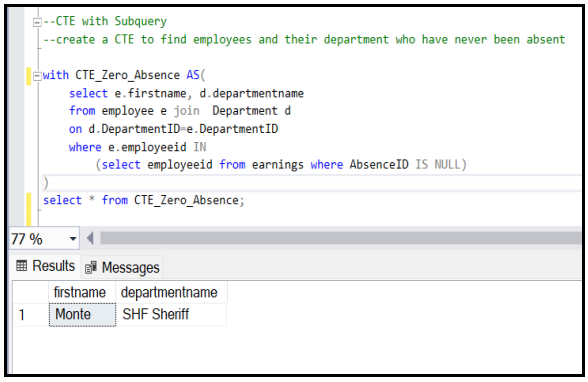

Fig. 17. SQL CTE for finding employees who were never absent

VIII. KEY INSIGHTS

The database analytics performed on the City Employee Earnings and Absences dataset revealed several important organizational trends.

- **Absenteeism Reduces Earnings**
Employees with high absenteeism show sharply reduced or zero earnings, confirming a strong negative link between attendance and compensation. This highlights absenteeism as a major factor affecting productivity.

- **Absences Follow Seasonal Patterns**
Absences peak in specific seasons. For example, some employees miss the most work in summer while others peak in spring or fall showing that attendance behavior is seasonal, not random.

- **Departments Show Uneven Performance**
  Departments like PPD Police and Streets generate the highest earnings, while Water, Parks & Recreation, and Health show lower totals, reflecting differences in workforce size, workload, and pay structures.

- **Retention Varies by Department**
  Some departments maintain stable workforces, while others experience high turnover. These patterns indicate where HR efforts are needed to improve retention.

- **Normalization Improved Data Quality**
  Converting the flat file into a normalized relational database removed redundancy, improved accuracy, and made analysis faster and more efficient.

- **SQL Features Added Automation and Insight**
  Triggers, stored procedures, window functions, and CTEs enhanced automation and analytical depth, supporting realistic payroll, attendance, and performance insights.

- **Absence Reasons Became More Useful**
  Adding descriptive absence information improved clarity and enabled better analysis between approved and unapproved leave types.

IX. CONCLUSION

This project successfully applied end-to-end SQL-based analytics to transform raw employee earnings and absence data into meaningful organizational intelligence. Through systematic data cleaning, normalization, relational modeling, and analytical querying, the study uncovered clear relationships between absences, earnings, departmental performance, and retention. The star-schema design enabled efficient multi-dimensional analysis, while SQL constraints ensured data integrity and accuracy. Advanced components such as triggers, stored procedures, and indexing demonstrated how database-level automation and optimization directly support operational decision-making.

The findings offer valuable guidance for organizational planning. High absenteeism negatively affects earnings and departmental output, indicating potential areas for policy review or targeted intervention. Departments with high turnover may benefit from improved retention strategies, while high-performing units can serve as benchmarks for broader organizational improvements. Overall, this work demonstrates the value of structured database analytics as a powerful tool for enhancing workforce management, improving departmental effectiveness, and supporting long-term organizational stability.

REFERENCES

[1] City of Philadelphia, "City Employee Earnings." *Data.gov*. Available: https://catalog.data.gov/dataset/city-employee-earnings

[2] UCI Machine Learning Repository, "Absenteeism at Work Dataset." *University of California, Irvine*. Available: https://archive.ics.uci.edu/dataset/445/absenteeism+at+work

[3] Lucid Software Inc., "Employee Database Conceptual Framework Diagram." *Lucidchart*. Available: https://lucid.app/lucidchart/18717c13-39e2-4846-bbce-64a6e48fbe75/edit?invitationId=inv_19f01b65-1529-48b0-adf9-86cedf10d90e

AUTHOR INFORMATION

Maksuda Elahi is currently studying Business Intelligence Systems Infrastructure at Algonquin College, developing strong skills in SQL, Python, and data visualization to interpret and transform complex data. Drawing on her experience in software QA, she brings precision and analytical depth to every project. She aims to create meaningful insights that support impactful decision-making in BI and data analytics.

Shara Khandakar holds a bachelor's degree in economics with a major in Finance and is currently pursuing a Graduate Certificate in Business Intelligence Systems Infrastructure at Algonquin College, Ottawa. Her interests include machine learning, data visualization, database systems, and business analytics. She aims to work in the field of data engineering and applied machine learning.

Murk Asad received M.Sc. degree in Information Technology from National University of Sciences and Technology (NUST), Islamabad, Pakistan, and a Postgraduate Certificate in Artificial Intelligence from George Brown College, Toronto. She has professional experience as a Business Analyst in the supply chain domain and is currently pursuing a Graduate Certificate in Business Intelligence Systems Infrastructure at Algonquin College, with academic and professional interests in data analytics, machine learning, and advanced database systems.

Anam Vakil holds a Master of Science in Information Technology and is currently pursuing Business Intelligence System Infrastructure at Algonquin College in Ottawa. With over three years of work experience as a Data Analyst, her career focus has been to develop skills within the IT field, which include business analytics, data visualization, database systems and machine learning. She aims to work in the field of data engineering and applied machine learning.