**Employee Management and Attendance System**

**1. Introduction**

This report outlines the development and implementation of a foundational Employee Management and Attendance System using Microsoft SQL Server. The project’s primary objective was to create a robust database structure capable of storing and managing employee information, department details, and daily attendance records. Beyond simple data storage, the system was designed to provide automated data integrity checks through triggers, enable complex data analysis via custom functions, and generate insightful reports for business intelligence purposes. This project demonstrates proficiency in database design, data manipulation, and the use of advanced SQL features to build a functional and scalable solution.

**2. Abstract**

The project involved the design and implementation of a relational database for an Employee Management and Attendance System. Four key tables were created: Departments, Roles, Employees, and Attendance. The database was populated with over 200 records of dummy data to simulate a real-world environment. A series of SQL components were then developed to enhance the system's functionality. Triggers were implemented to automatically manage timestamps and update attendance statuses, ensuring data accuracy. A custom SQL function was created to calculate total work hours, a key metric for performance analysis. Finally, a set of advanced queries utilizing GROUP BY and HAVING clauses was developed to generate detailed reports on monthly attendance, late arrivals, and departmental performance. The project successfully created a comprehensive system that can be used for effective human resource data management and reporting.

**3. Tools Used**

* **Microsoft SQL Server:** The relational database management system (RDBMS) used for all data storage and management.
* **SQL Server Management Studio (SSMS):** The graphical user interface (GUI) tool used to connect to the SQL Server, write and execute queries, and manage database objects.
* **T-SQL (Transact-SQL):** The proprietary extension of SQL used within Microsoft SQL Server to create all the database schema, triggers, functions, and queries.

**4. Steps Involved in Building the Project**

The project was executed in a systematic, step-by-step manner to ensure a logical and functional final product.

**Step 1: Database and Table Creation** The first step involved defining the database schema. Four tables were designed with appropriate columns, data types, and relationships.

* **Departments:** Stores department names.
* **Roles:** Stores job titles or roles.
* **Employees:** Stores employee-specific information and links to Departments and Roles.
* **Attendance:** Records daily sign-in and sign-out times for each employee. Foreign keys were established to enforce referential integrity between the tables.

**Step 2: Data Insertion and Population** To simulate a live environment, dummy data was inserted into all tables. Over 200 employee records were programmatically generated to demonstrate a large dataset. A T-SQL script was written to loop through all employees and generate three months' worth of random daily attendance records, including instances of present, absent, and late arrivals.

**Step 3: Query Development for Data Analysis** Initial queries were written to extract meaningful information from the data.

* **Monthly Attendance:** A query was developed to summarize attendance by month, providing a clear count of present, absent, and late days for each employee.
* **Late Arrivals:** A second query was created to specifically identify all instances of late arrivals, listing the employee, date, time of sign-in, and minutes late.

**Step 4: Automation with Triggers** To automate data management and ensure integrity, three triggers were created on the Attendance table.

* **trg\_Attendance\_Insert\_Timestamps:** Automatically sets the CreatedAt and UpdatedAt timestamps when a new record is inserted.
* **trg\_Attendance\_Update\_Timestamp:** Automatically updates the UpdatedAt timestamp whenever an existing record is modified.
* **trg\_Attendance\_SetStatus:** Automatically checks the SignInTime and sets the Status column to 'Present' or 'Late' without manual input.

**Step 5: Custom Function Creation** A scalar-valued function, dbo.CalculateTotalWorkHours, was created. This function accepts an EmployeeID, a StartDate, and an EndDate as parameters. It calculates the total minutes worked within the specified period and returns the sum converted to hours, providing a reusable and efficient way to query for this metric.

**Step 6: Report Generation** Finally, advanced reports were generated using GROUP BY and HAVING clauses to filter and summarize aggregated data.

* **Department Performance Report:** This report calculates the average work hours per employee per day for each department and filters the results to show only those with an average of more than a certain threshold, helping to identify high-performing departments.

**5. Conclusion**

This project successfully demonstrates the comprehensive use of Microsoft SQL Server for building a functional and robust employee management system. By leveraging T-SQL features such as triggers, functions, and advanced reporting queries, the system achieves a high level of automation and data integrity. The implemented solution is not just a data repository but a powerful tool for analyzing workforce attendance and productivity. The modular design of the tables and components ensures scalability, making it a solid foundation that can be expanded with additional features such as payroll integration, leave management, or performance tracking in the future.