**Create a Visualization in Snowflake Snowsight**

1. **Abstract:**

In today's data-driven landscape, effective visualization of data plays a crucial role in understanding trends, patterns, and insights that drive informed decision-making processes. Snowflake Snowsight, a powerful data visualization tool integrated within the Snowflake cloud data platform, empowers users to create interactive and insightful visualizations directly from their data stored in Snowflake's data warehouse. This project focuses on leveraging Snowflake Snowsight to design and implement compelling visualizations that enable users to explore and analyze data intuitively.

Unlike Hive, which requires managing a Hadoop cluster with complex infrastructure setups, network security, and ongoing maintenance, Snowflake is fully managed in the cloud, eliminating the need for users to handle hardware, scaling, or infrastructure management. Hive users must ensure their clusters are secure and maintain them even after the project ends, leading to additional costs and administrative effort. In Snowflake, all resources are managed automatically—compute can be scaled up or down as needed and stops when not in use, meaning users only pay for what they consume. This makes Snowflake more efficient, secure, and cost-effective compared to Hive’s Hadoop-based architecture.

| **Existing System** | **Proposed System** |
| --- | --- |
| Existing system are Hive and RDBMS | Proposed System is Snowflake |
| Requires complex setup and ongoing maintenance | Fully managed in the cloud with automated updates |
| Limited by cluster size (Hive) or vertical hardware limits (RDBMS) | Elastic scaling for both compute and storage |
| Slow for real-time analytics and requires manual tuning | Fast real-time analytics with automatic query optimization |
| High fixed costs for running clusters and infrastructure | Pay-as-you-go model; only pay for what you use |
| Difficulties handling semi-structured data and limited to structured data | Seamlessly manages structured and semi-structured data |
| Manual security setup and management | Built-in security features with automated compliance |
| Requires technical knowledge for setup and query tuning | User-friendly interface with easy visualizations |
| Struggles with high user concurrency and requires external tools for processing | Supports high concurrency with native data integration |

| **Existing Algorithm** | **Proposed Algorithm** |
| --- | --- |
| * **Hive**: MapReduce, Tez/Spark, partitioning. * **RDBMS**: ACID transactions, indexing, various join algorithms, query optimization. | **Snowflake:**   * Cost-based Query Optimization * Automatic Clustering * Massively Parallel Processing (MPP) Algorithm * Micro-partitioning * Multi version Concurrency Control (MVCC |

**iii.**

**iv.**

| **Algorithm Definition** | **Algorithm Definition** |
| --- | --- |
| **MapReduce**: Distributed processing algorithm for large-scale data, splitting tasks into Map and Reduce phases.  **Tez:** Optimized task execution using Directed Acyclic Graphs (DAG), improving efficiency over MapReduce.  **Spark**: In-memory data processing for faster execution, also using DAG for task scheduling.  **ACID** **Transactions**: Ensure Atomicity, Consistency, Isolation, and Durability in database operations.  **Indexing**: A data structure (e.g., B-tree) that improves the speed of data retrieval by organizing database records for faster searches.  **Query** **Optimization**: The process of selecting the most efficient way to execute a SQL query using cost-based or rule-based optimization strategies. | **Cost-based Query Optimization**: Evaluates various query execution plans and selects the most efficient one based on estimated costs.  **Automatic Clustering**: Automatically reorganizes data storage to optimize query performance without requiring manual tuning.  **Massively** **Parallel** **Processing** (**MPP**): Utilizes multiple processors to perform tasks concurrently, significantly speeding up query execution.  **Micro**-**partitioning**: Divides large datasets into smaller, manageable segments to enhance query performance through efficient data scanning.  **Multi-Version** **Concurrency** **Control** (**MVCC**): Maintains multiple versions of data to allow concurrent transactions without conflicts, ensuring data consistency. |

**v.**

| **Drawbacks of Existing System** | **Advantages of Proposed System** |
| --- | --- |
| Slow Query Performance | Fast Query Execution |
| Manual Tuning Required | Automatic Clustering |
| Limited Concurrency | High Concurrency |
| Scalability Issues | Seamless Scalability |
| Data Duplication | Zero-Copy Cloning |
| Complex Setup and Maintenance | Fully Managed Service |
| Limited Support for Semi-Structured Data | Support for Semi-Structured Data |
| Data Ingestion Latency | Real-Time Data Processing |

**vi. MINIMUM SYSTEM REQUIREMENT**

## **Hardware Requirements:**

* Processor: DUAL CORE 200
* Ram: 2-4GB DDRAM
* Hard disk: 29GB

# **Software Requirements:**

* IDE: ECLIPSE
* Operating system: WINDOWS 7/10.

**Network requirements:**

* Need a valid Snowflake account with the required access and permissions
* Good internet connection and bandwidth