# **Business Intelligence Engineer Notes**

## Data Warehousing Concepts:

Storing large amount and complex data in an organized way. Jaise Almera mei different section hote hai waise he data ko different places pe store kiya jaata hai based on certain criteria so that it can be accessed and analyzed easily.

How it works:

**Data collection**: Data is collected from different source (sale transaction, customer interaction, websites) , these data are structure(table) and unstructured (images, text) both.

**Data integration:** all collected data combined at one place

**Data Storage:** Integrated data stored in data warehouse (which has different drawer)

**Data organization:** data is organized (such as structuring the data into tables)

**Data analysis:** Analysed to uncover the pattern and trends

**Reporting and visualization**

**Question1: What Issues occur with building the Warehouse?**

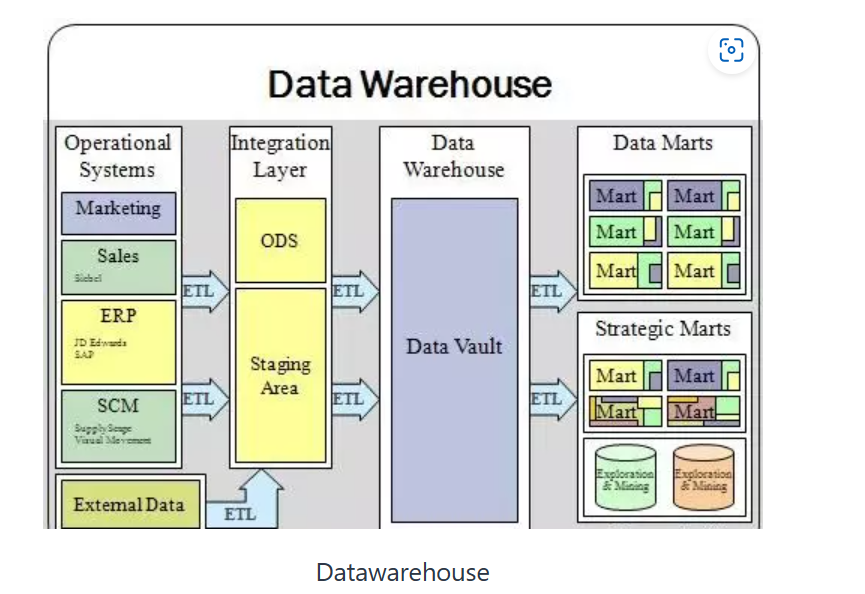
1. **When and how to gather data (continually or periodically)- To** keep warehouse up to date it should be sync with the updated source via 2 phase commit, which generally very expensive.
2. **What schema to use?** Data sources are constructed independently and so they are likely to have different schema. Thus, these schemas are integrated before storing to warehouse and therefore they are not the exact copy of the data at the source.
3. **Data transformation and cleaning**
4. **How to propagate updates?** Updated become problem when relation at data warehouse is not same as that at source
5. **What data to summarize?**

**Question 2: Data Warehouse vs DBMS**

| **Database** | **Data Warehouse** |
| --- | --- |
| A common Database is based on operational or transactional processing. Each operation is an indivisible transaction. | A data Warehouse is based on analytical processing. |
| Generally, a Database stores current and up-to-date data which is used for daily operations. | A Data Warehouse maintains historical data over time. Historical data is the data kept over years and can used for trend analysis, make future predictions and decision support. |
| A database is generally application specific.  Example – A [database](https://www.geeksforgeeks.org/what-is-database/) stores related data, such as the student details in a school. | A Data Warehouse is integrated generally at the organization level, by combining data from different databases.  Example – A data warehouse integrates the data from one or more databases , so that analysis can be done to get results , such as the best performing school in a city. |
| Constructing a Database is not so expensive. | Constructing a Data Warehouse can be expensive. |

**Disadvantage of data warehousing:** Cost, Complexity, time consuming, Data integration challenges, data security

***Stages of datawarehousing***



***What are the steps to build the datawarehouse?***

Following are the steps to be followed to build the datawaerhouse:

* Gathering business requirements
* Identifying the necessary sources
* Identifying the facts
* Defining the dimensions
* Defining the attributes
* Redefine the dimensions and attributes if required
* Organize the Attribute hierarchy
* Define Relationships
* Assign unique Identifiers

## **Dimensional Modelling (Start and Snowflake schema)**

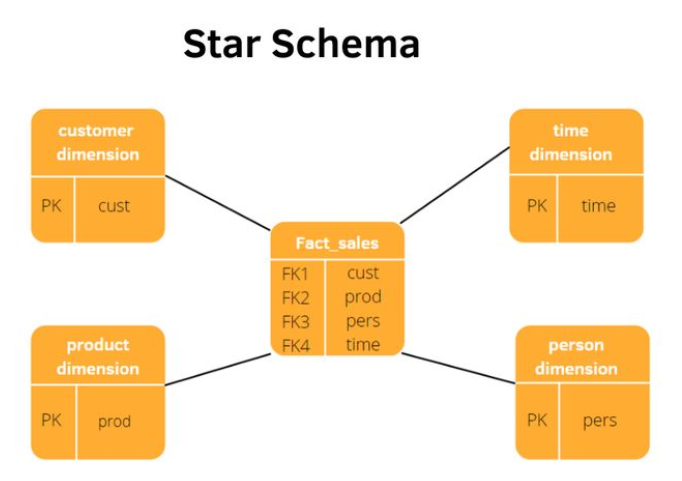
Dimension modeling is a concept used by data warehouse designer to build their own data warehouse.

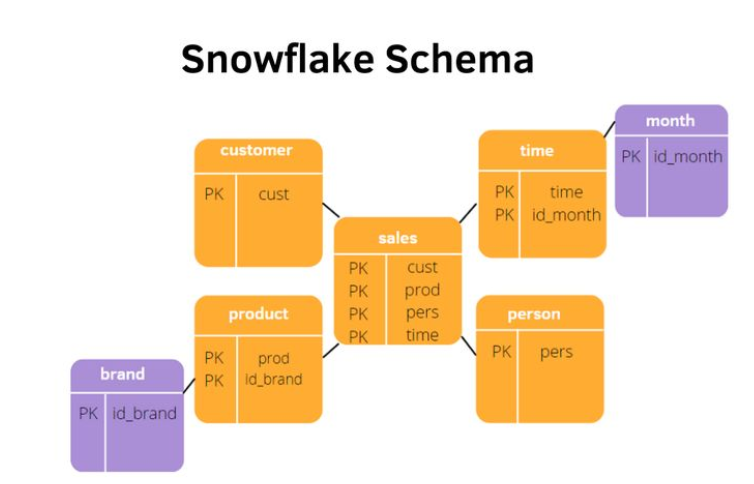
Purpose: It facilitates the querying, reporting and analysis of data stored in warehouse.

There are 2 schema for dimension modelling: Start and snowflake

This model can be stored in 2 types of tables: Facts and Dimensions table.

**Fact table has facts and measurements of the business (quantitative data- revenue, quanity sold, expense) and the dimension table contains the context of measurements (non-quantitative data like time, geography, product, customer).**

****

****

**Snowflake schema:** extension of start schema, when dimension tables are normalized into multiple dimension table, this schema save storage space and improve data consistency but increase the complexity.

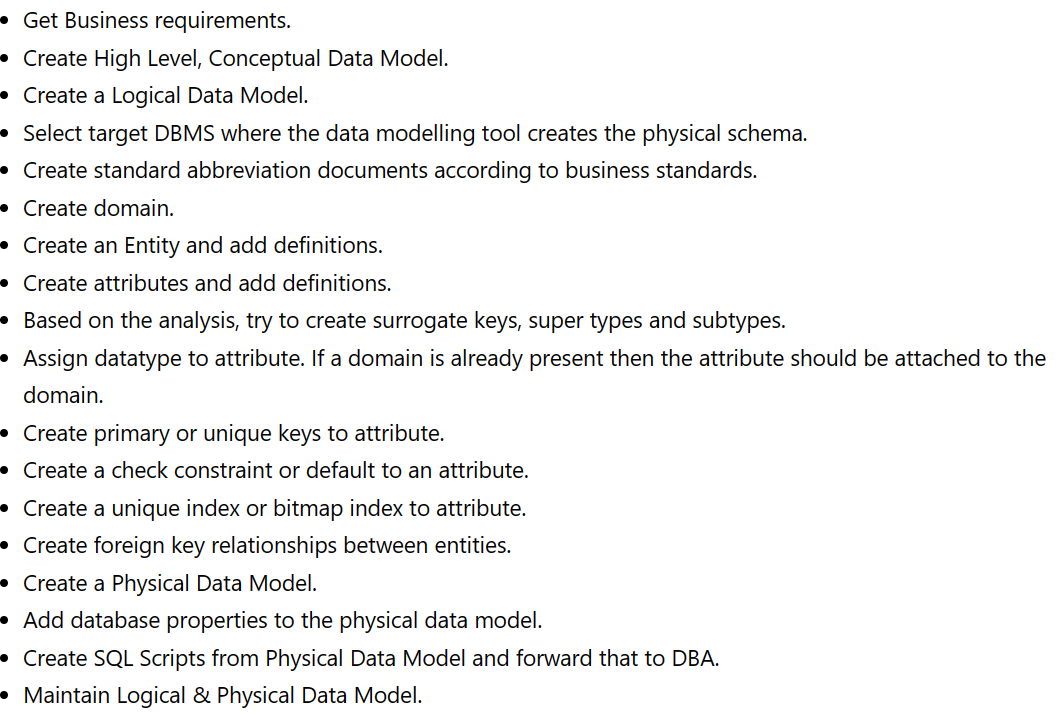
**Hierarchy:** Dimension table often has a hierarchy where attributes are organized into levels of detail. For example, a time dimension might have levels like year, quarter, month, and day.

**Slowly Changing Dimensions (SCDs):** change over time but at a relatively slow rate. Examples include customer addresses, product categories, etc.

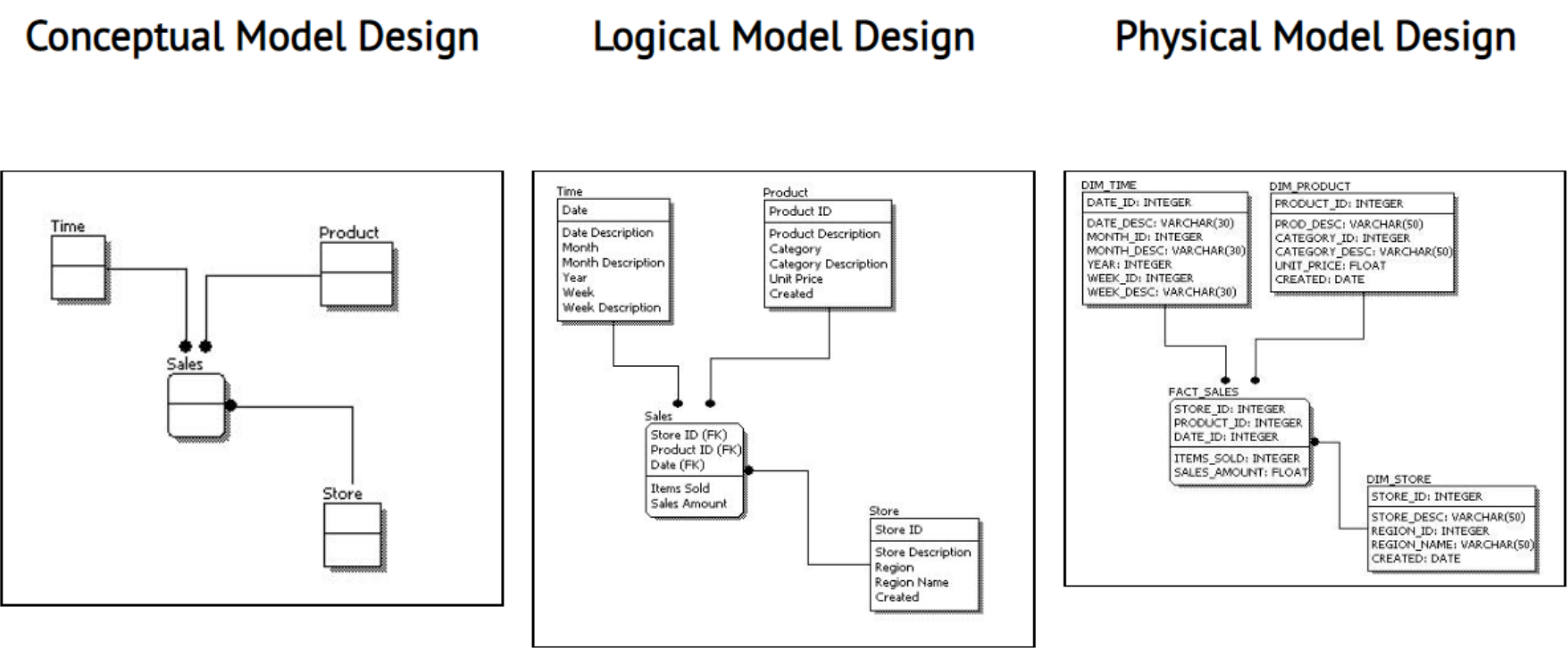
Dimensional modeling techniques include methods for handling SCDs, such as **Type 1** (overwrite), **Type 2** (add new records), and **Type 3** (add new columns).

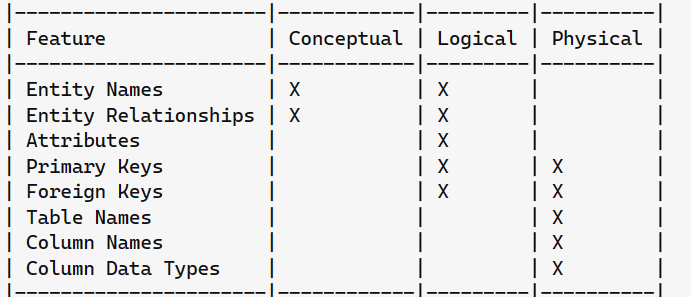
**Factless Fact Table:** contain no measures but are used to represent relationships between dimensions

**Steps to create data model:**

****

**Surrogate Key:** A surrogate key is a unique identifier assigned to each record in a table, typically for the purpose of uniquely identifying rows, especially in cases where there is no natural key or the natural key is not suitable for this purpose.

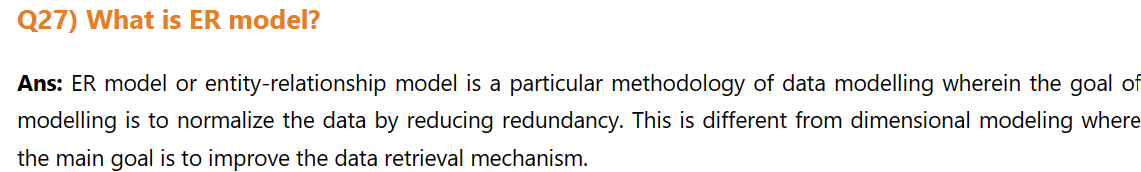
****

****

1. Conceptual – the “what” model
2. Logical – the “how” of the details
3. Physical – the “how” of the implementation

# When should we consider denormalization?

When there are a lot of tables involved in retrieving data.



# ETL (Extract, Transform and Load)

Extract: Extracting data from heterogenous rescources (db, files, apis, web services) could be transactional system- erp,crm, social media. Thos extraction can be incremental (only new or modified data after last extraction) or full (entire dataset)

Transform: cleaning, enriching(adding new calculated field), aggregating, data validation, restructuring(normalization)

Loading: transforming data to target repository (data warehouse, data mart or data lake). It can be performed through bulk loading, incremental loading or real-time loading. Loading operation should be optimized for performance and data integrity.

## Interview questions on ETL

1. **What are the common data transformation tasks are performed during ETL process.**

**Ans: Data cleaning, data validation, data normalization, data aggregation and data enrichment and data handling issues.**

**Data cleaning** include removing duplicates, replacing missing values and removing errors.

**It involves data profiling, data validation, data standardization, deduplication(removing duplicate records), data enrichment(**adding additional information as per the business requirement) **and monitoring.**

**Data profiling:** Process of analysing and understanding the data pattern, distribution and relationships

**Data validation** include checking the completeness of data, accuracy, consistency, validity (as per the given criteria), integrity (checking foreign key relationship are valid and consistent, timeliness(verifying up-to-date data as asked)

**Some of the examples for data validation- test cases:**

1. Checking the data structure: describe table\_name (will give the information of table structure)
2. Check the total count in source and target (should be same): select count(\*) from source/target
3. Verify the duplicate records: select count(\*), emp\_id, name from source group by empid, name.
4. Verify source and target data: select \* from source except select \* from target (will check if there is record which is present in source but not in target) UNION

select \* from target except select \* from source (will cheack if there is any record in target which is not in source)

1. Verify transformation logic.

**Data standardization**: process of converting data into standard format( checking unit of measurement, naming covension, formatting)

1. **Staging area in ETL?**

A temporary repository for data before it goes for transformation, it contains raw data extracted from different sources. It allow the cleaning, merging, and reformatting of data

1. **What factors should be considered when designing the ETL process for performance optimization?**

Factors include data volume, data complexity, hardware resources, network bandwidth, parallel processing, indexing, data partitioning, and caching strategies.

**How we do that**:

1. **optimize SQL query** (minimize the use of sub query, selecting only the column you need, use wildcard only at the end of phrase, use limit top preview the query result, avoid use of distinct if possible, replace UNION with UNION ALL, run large query during off-peak hour), Use indexing (example, to search salary where 10000, use of index with divide salary into range of buckets, it use btree for thus and retrieval is fast in this case)
2. **Parallel Processing:** Partioning the data into smaller set that can be processed in parallel
3. **In-**memory processing: Holding data in RAM, caching mechanism store intermediate result so that repeated calculation are not necessary.
4. **Batch processing or micro-**batching (processing smaller set of data at regular intervals).

# **Data mart and Data warehouse**

**Data mart:**A data mart is a subset of a data warehouse that focuses on a specific business area, department, or subject area.

**How do you decide whether to use data mart or data warehouse**

The decision to use a data mart or a data warehouse depends on factors such as the scope of analysis, the level of detail required, the specific needs of end-users, and the organization's overall data strategy. Data marts are typically suitable for departmental or business-specific analysis, while data warehouses are designed for enterprise-wide reporting and analysis.

# **Data Lake and their integration with BI**

# Difference between data lake and Data warehouse

The data lake is a centralized repository for storing unstructured, structured, and semi-structured data in raw form and unlike Data warehouse it doesn't require predefined schema and data transformation before storing data.

**Challenges** **organizations** **face** **when** **integrating** **data** **lake** **with** **BI** **tool**

The decision to use a data mart or a data warehouse depends on factors such as the scope of analysis, the level of detail required, the specific needs of end-users, and the organization's overall data strategy. Data marts are typically suitable for departmental or business-specific analysis, while data warehouses are designed for enterprise-wide reporting and analysis.

# **SQL server (Performance tuning and optimization)**

Clustered index:

An index is a disk-based structure linked to a table to facilitate quick response.

Cluster index can be performed when data is sequential and sorted order . If you have atable that already has primary key then automatically that primary key become cluster index. Agar primary key ki value 1 to 100 hai toh clustering btree k hisab se half half tree jaisi banti hai so that retrival fast ho.

Non-clustering index: is like index page of book, so here data is stored in one place and clustred index is stored in other place.

Query to write for creating non-cluster index

create nonclustered index NIX\_FTE\_Name on Student (Name ASC);

***When to use them:***

### **Index Clustering**

* Perfect for tables where range queries, in particular, place a high value on data retrieval efficiency.
* Ideal for tables with few updates or relatively static data because moving data around can slow down insert and update operations.

### **Non-Clustered index**

* allows for the optimization of various query types without changing the data’s physical order on disk.
* Ideal for tables where data changes often since inserts and updates are typically quicker than with clustered indexes.

interview questions tailored for a Business Intelligence (BI) role focusing on performance tuning and optimization techniques:

1. **Can you explain the importance of performance tuning in the context of Business Intelligence systems?**

Enhance user experience, improved efficiency, increased productivity, scalability, better decision making, customer satisfaction

1. **What are some common performance bottlenecks encountered in BI systems, and how do you approach identifying them?**

Some of the common bottleneck issues are: Slow query execution, insufficient data model, resource contention, inadequate indexing, data skew, network latency.

To identify these bottleneck , below approaches use:

**Query profiling:** with the help of SQL server profiler, we can identify as which query is taking how much time and how it can be changed (as suggested under tuning template of profiler) to improve it by x%.

**Data model analysis:** review the model schema and indexing strategies and improve it by denormalizing or adjusting index

**Resource monitoring:** check the memory consumption and area of improvement

**Network performance monitoring:** check the network latency and throughput between bi server and data source.

**Index usage analysis**

**Data distribution analysis**

--------------------------------------------------

SQL performance tuning can be divided into 3 level:

* 1. Query level optimization (check the index usage, partition usage, select column -list, split complex query, check join condition)
  2. Table (DDL) level optimization (check proper data type attached to each column or nit, compression is enabled or not, proper index column)
  3. Cluster level optimization (sufficient memory or node is a available or not, node types are appropriate or not, concurrent job during job run)

1. Could you discuss a specific instance where you identified a performance issue in a BI system and the steps you took to resolve it?
2. When optimizing the performance of a BI query, what factors do you consider, and what techniques do you employ?

**Factors to consider:**

**Query complexity**: number of ables, size of the dataset, joins complexity and filter

**Data model structure**: check schema and indexing strategy, do denormalization if possible

**Resource utilization:** memory space consumption

**Query execution plan:** how the database engine processes the query and identify potential areas for optimization, such as suboptimal join order or inefficient index usage.

**Data distribution:** rebalance the data across the partition and node

**Techniques:**

Indexing, Query rewriting, Partitioning, query optimization, **caching** [Implement query caching mechanisms to cache frequently executed queries], parallel processing

1. How do you prioritize performance optimization efforts when dealing with multiple performance issues in a BI environment?
2. Can you explain the role of indexing in optimizing database performance for BI queries? What considerations should be taken into account when creating indexes?

Role: fast retrieval of data, Improved query execution, enhance scalability

Considerations:

**Query pattern:** Apply index on column which has condition like, where or joins or group by clause is used

**Data distribution:** column which has wide range of distinct value, indexing is effective

**Index size:** evaluate the storage requirement of index and performance

**Maintenance overhead:** cost of index creation, updates, and rebuilds.

**Index type:** cluster ornon-cluster or composite or filter index to use

1. In a scenario where a BI dashboard or report is slow to load, what steps would you take to diagnose and improve its performance?

**Identify the Performance Issue: check which** aspect of the dashboard or report that is slow to load

**Review Dashboard Design and Layout**

**Analyze Data Sources and Queries:** query profiling, caching mechanism etc

**Optimize Data Loading and Refresh Times:**

1. What strategies do you use to optimize data extraction, transformation, and loading (ETL) processes in a BI environment?

**Incremental Loading:** Extract and load only changed or new data to reduce processing time.

**Parallel Processing:** Execute ETL tasks concurrently across multiple nodes for faster processing.

**Data Partitioning:** Divide large datasets into smaller subsets to enhance processing efficiency.

**Batch Processing:** Group ETL tasks into batches to optimize resource usage.

**Optimized Data Structures:** Use optimized schemas and storage formats for efficient transformations.

**Indexing and Materialized Views:** Create indexes and precomputed aggregates for quicker data retrieval.

1. How do you ensure that BI reports and dashboards remain performant as data volumes and **user concurrency increase over time?**

**Continuous Monitoring, Scalable Architecture, Query Optimization, Caching Mechanisms, Load Balancing, Resource Allocation:**

1. Can you discuss the impact of hardware resources (e.g., CPU, memory, storage) on the performance of BI systems and any best practices for optimizing hardware configurations?
2. When dealing with slow-performing SQL queries in a BI system, what are some optimization techniques you might employ?
3. How do you balance the trade-off between optimizing query performance and maintaining data accuracy and consistency in BI reporting?
4. What role does caching play in improving the performance of BI queries and reports, and how would you implement caching strategies in a BI environment?
5. Could you share your experience with implementing parallel processing or distributed computing techniques to improve BI query performance?
6. In your opinion, what are some emerging trends or technologies in BI performance optimization that professionals in this field should be aware of?