sisense

WHITEPAPER

How to Meet the Demand for Analytics Across the Organization

A new architecture for breakthrough analytics performance and clarity

Table of Contents

A new architecture

Putting analytics in the hands of everyone is now critical to business performance

Forrester Research recently noted in The State of The Insights Driven Business, 2022, that companies driven by insights are eight times more likely to say they grew by 20% or more than firms with less analytics maturity. **And with the current economic and business uncertainty, analytics are more critical than ever.**

Yet analytics are only as good as the user engagement they receive. And in a world where multi-second response times can lose an audience, performance is crucial. The simple fact is that performance is critical for user engagement.

Success is not only measured in terms of how quickly a query turns into results but also the sustainability to achieve it— the effort behind the scenes to deliver performance and maintain it as the data, dashboards, and users' scale.

Today's self-service users have incredibly high expectations and are unaware of the resources and effort required behind the scenes. Data visualization, query, and dashboard technology have struggled to keep up with the data volumes and increasingly sophisticated requirements. Data governance and security are subject to more governance and security than ever before.

One of the hard truths in analytics is that ways to improve performance haven't changed in decades—

tune the data model and database, cut the data volumes, add more processing resources, or optimize the query. They all come with their own compromises, whether the direct cost of adding more infrastructure or the indirect costs of adding care and feeding in terms of database and SQL experts. And worse, with increasing data model and schema optimization, SQL queries become harder to understand, harder to optimize, and ultimately harder to trust.

There aren't enough SQL, database, and analytics specialists to keep ever-growing analytics demands in check. It's time for a better way.

This whitepaper explores today's analytical performance challenges that analytics deployments everywhere are facing—**and how Sisense is innovating to traditional break to enable teams to extend analytics to everyone at scale.**

Achieving sustainable analytics

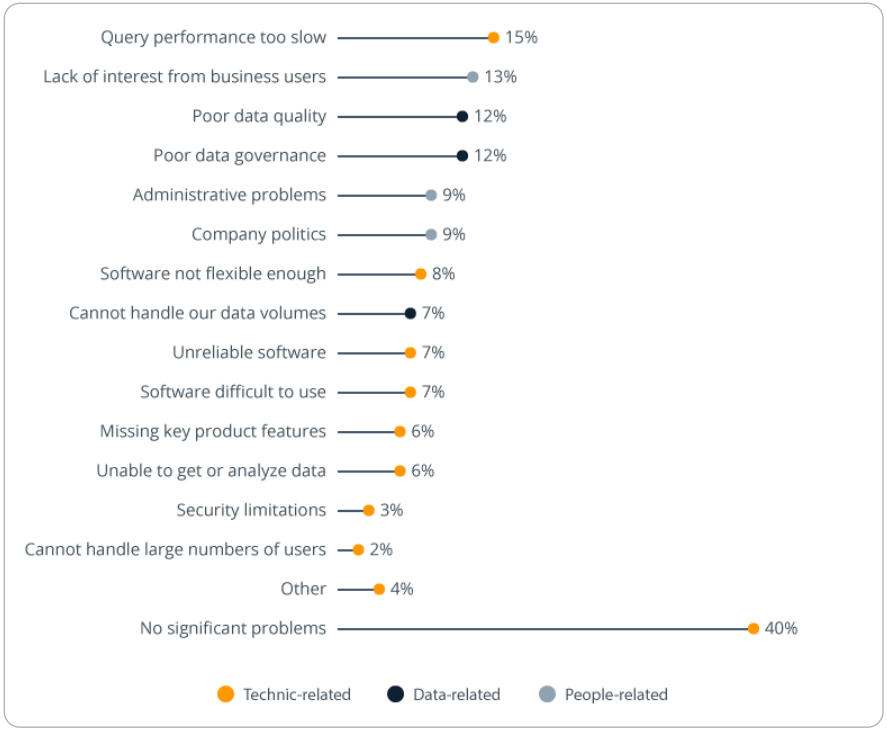
Why sustainable performance? With an unlimited budget, limitless resources, and time, anyone can achieve performance with analytics.

The problem is that few or no organizations have that luxury.

Ultimately, throwing people—DBAs, SQL and analytics specialists, and infrastructure, to keep an analytics deployment running responsively is not sustainable or ultimately feasible.

It's precisely for this reason that when an analytics deployment runs into a problem, the top reason cited is often that query performance is too slow—whether it's latent dials on a dashboard or long-running reports.

That was precisely one of the conclusions of the '22 BARC survey on The Most Common Business Intelligence Problems, one of the most extensive surveys on why BI deployments fail.



The most frequent problems in BI projects, BARC, 2022

Perhaps unsurprisingly, the second most significant reason for failure is business users' lack of interest. In fact, in many cases, poor performance and lack of user engagement go hand in hand.

Poor performance—like waiting for dashboards or dashboard components to load or reports to run hurts everyone. Frustration amongst data consumers stuck waiting too long for results often means they'll disengage, heading back to spreadsheets. That's a death knell for analytics success—whether you deliver analytics to your team or embed it in your product.

Traditional compromises

The ways to achieve performance haven't changed in years, and all come with compromises. Many analytics platforms rely on SQL or database expertise to care and feed the deployment.

For those in charge of deployment, the dashboard designers, database architects, and data modelers, it's a painful balancing act. For example, heavy data volumes, dashboard complexity, high concurrency of users hitting the same dashboards or charts, or data models with significant amounts of governance and row-level security applied to them can all create issues.

Some of the most common approaches to tuning performance include:

1. Aggregate the data in your data warehouse.

One method is aggregating the data, for example, summarizing it at the hourly or daily level of granularity in the underlying database, whether it's Redshift, BigQuery, Snowflake, or another DB. This drastically reduces the number of rows of data and will typically increase performance. However, this often creates limitations in how far the analyst can drill down into the data in the data model, or they'll have to drill across to a separate set of tables that contain the detail, as well as having to create two sets of data (the detail, and the aggregated tables).

2. Reduce your dataset

Similar to aggregation, reducing data is also an option, such as removing extraneous dimensions from the data model or eliminating tranches of historical data. While this may provide near-term performance improvements, the increasing need to eliminate and manipulate the underlying data to keep analytics at low latency becomes increasingly challenging over time and is ultimately not sustainable.

All of this has traditionally often required a substantial amount of database work and often requires some technical skills. And as analytics requirements and dashboards change, this can create significant ongoing overhead.

3. Add more infrastructure

Another option is to spend more on your server and database infrastructure, whether adding more memory and CPU to reduce page faults or better detail with concurrent quests, adding more clusters, or increasing the cluster size. However, ultimately, this is not sustainable simply from a cost perspective.

4. Focus on tuning and database caching

Most modern databases support caching, where they cache the results of certain types of queries in memory. When a user submits a query, the database checks the results cache for a valid, cached copy of the query results. If a match is found in the result cache, it uses the cached results and doesn't run the query. In some cases, the cache size is tunable. However, a cache only gets you so far. If the queries are unpredictable, such as with ad hoc analysis, cache hits will be low.

5. Optimize the schema and data model.

Optimize the schema and data model. Other options often include radically de-normalizing the data model, optimizing parts of the schema for queries, clustering, partitioning, or other options. Unfortunately, these often become the law of diminishing returns as data and queries grow. Optimizing the data model also often comes with the additional overhead of increased query complexity—with increasingly indecipherable SQL that risks inserting erroneous results into a dashboard or report.

SQL conundrum

Optimizing the data model and ever-increasing data level governance and security come with a heavy price: **query complexity.**

SQL becomes increasingly hard to read, understand, and tune as the data model and database deployment become increasingly esoteric.

Ultimately, it often adds to the pain of sifting through queries, hunting for ways to optimize, or simply dialing back dashboard expectations and use cases to fit within practical constraints.

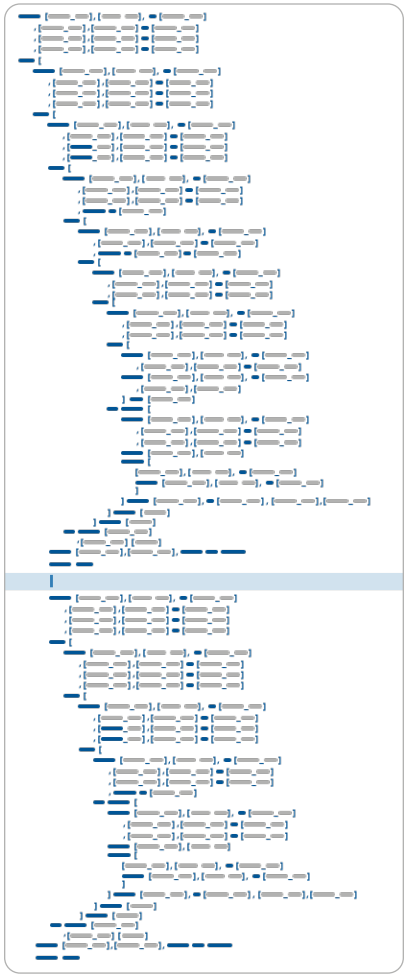
The problem is that sifting through the SQL queries generated by some analytics tools can be like trying to decode hieroglyphics. While manually optimizing the SQL with complex data models gets torturous, quickly becoming hard to read, debug, and test—risking the integrity of results.

Analytics teams should focus on rolling out more dashboards, insight, and value, not buried in query tuning, testing, and debugging.

The goal is to achieve strong query performance to delight consumers and drive engagement without bringing the analytics operations team—those in charge of delivery, to their knees in never-ending manual query optimization and trying to decode obtuse SQL statements.

Many organizations lack the deep in-house SQL skills to understand the kind of queries being constructed by consumers and where there are opportunities to optimize them based on the data model. In addition, the proper sorting, aggregations, different join operations, and execution order can significantly impact performance, and it gets complex fast.

This is compounded by the fact that there are so many data warehouses, each with its SQL nuances, whether Snowflake, Redshift, BigQuery, Azure Synapse, Databricks, or a host of other platforms. Each can benefit from solid query optimization to take advantage of their native dialects and query planning. Still, the truth is that many analytics tools don't do it well enough—and tuning for database-specific queries can be error-prone.



Analytics performance mandate

Demand for analytics across the organization is set to grow at an incredible pace, with IDC recently predicting that big data and analytics software adoption will more than double by 2026. IDC notes that **"demand for decision support, decision augmentation, and decision automation technology remains very strong despite short-term macroeconomic and geopolitical headwinds."**

With analytics adoption set to increase so sharply, it's time to rethink how to achieve sustainable performance—that enables organizations to achieve robust performance at scale without adding cost, overhead, or query complexity.

Nirvana is to make achieving incredible analytics performance incredibly simple. But it's also critical not to compromise readability and transparency in the process so that business teams are confident in the results.

Success is at the nexus of easy-to-achieve query performance and clear query clarity and transparency, while ensuring robust security and governance.

Introducing the Analytical Engine

Pioneered by Sisense, the Analytical Engine is a new way for organizations to achieve performance and clarity at scale. It's like your analytics platform shipping with your personal SQL wizard—who'll ensure the query you ask is easy to understand and turn it into the best SQL possible behind the scenes by executing a series of innovative, continually improving optimization steps.

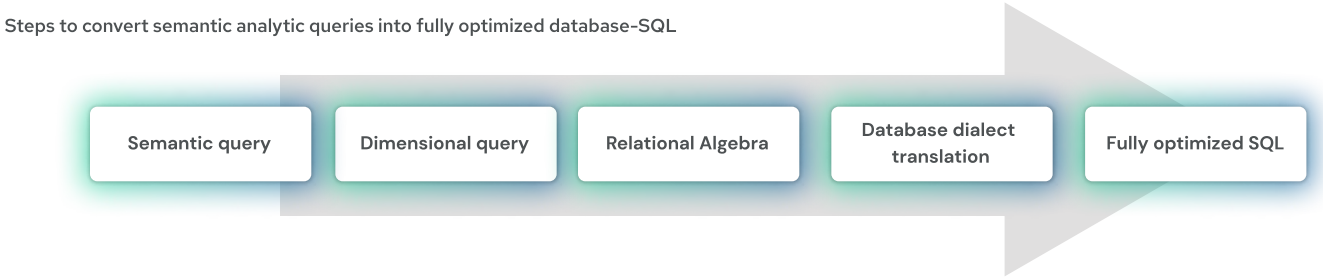
The goal is to unlock the full power of your data warehouse and model completely transparently to analytics consumers.

First, let's cover the design axioms and what we set out to build. An Analytical Engine focuses on three key areas, performance, query clarity, and security:

- **Effortless performance -** Using advanced relational algebra techniques, sophisticated dialect translation, cues from the analytics the user is running, and advanced query planning, an Analytical Engine generates highly efficient queries that take advantage of all the performance of the specific underlying data warehouse.

- **Query clarity -** Many data discovery tools turn user questions into nearly impossible-to-read queries. Often the query they generate looks nothing like the results they are looking for, requiring experts to decipher them. In contrast, the design goal of an Analytical Engine is to create straightforward, readable queries and optimize them behind the scenes for every database, making validation, verification, and testing easier.

- **Security intelligence -** Advanced data security often comes with a steep performance price tag, adding additional layers of query logic that saps performance. An Analytical Engine uses specialized relational algebra to keep the correct data flowing to each user geared to data-level security requirements without dragging down performance.



Conceptually, an Analytical Engine consists of multiple layers, starting with the semantic query:

1. Semantic query

A semantic query is one that clearly represents the business results the user is looking for from the dashboard. Without all the complexity of the data model, it clearly expresses the dimensions in play, metrics, and calculations as a declarative query. The goal here is clarity and understandability for the business user.

2. Dimensional query

Consuming the semantic query, the first step in formulating a highly-optimized query is transforming it into a dimensional graph-based query. The goal here is to converge the context of the semantic query the user is looking for with the realities of the underlying data model and apply a series of optimizations around areas such as time series, complex formulas at play, ranking, data types, and security filters.

3. Relational Algebra

With a fully-optimized dimensional representation of the query, turning it into logical database-agnostic Relational Algebra describes the entities, relationships, joins, fields, and other operations for the query—ultimately, an algebraic definition of the query that is optimized for the schema.

4. Database dialect translation

The final step is to produce execution-ready SQL highly optimized for the specific dialect of the database and in play, whether RedShift, BigQuery, Snowflake, Synapse, or another database.

Architecture of the Engine

In Sisense, the Analytical Engine is seamlessly embedded into the query flow, starting with the dashboard and reports analytics experience through to the resulting query execution against the storage engine in play.

First, let's cover the design axioms and what we set out to build. An Analytical Engine focuses on three key areas, performance, query clarity, and security:

- **Effortless performance -** Using advanced relational algebra techniques, sophisticated dialect translation, cues from the analytics the user is running, and advanced query planning, an Analytical Engine generates highly efficient queries that take advantage of all the performance of the specific underlying data warehouse.

- **Query clarity -** Many data discovery tools turn user questions into nearly impossible-to-read queries. Often the query they generate looks nothing like the results they are looking for, requiring experts to decipher them. In contrast, the design goal of an Analytical Engine is to create straightforward, readable queries and optimize them behind the scenes for every database, making validation, verification, and testing easier.

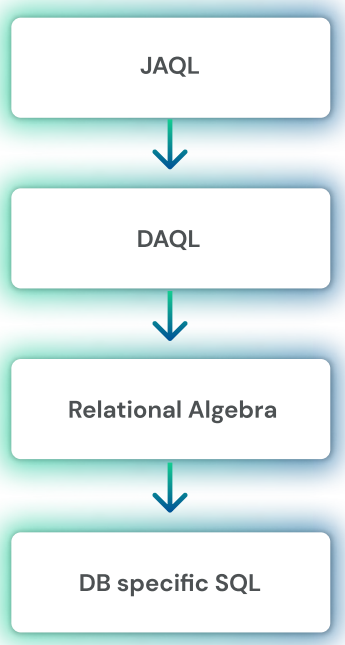
- **Security intelligence -** Advanced data security often comes with a steep performance price tag, adding additional layers of query logic that saps performance. An Analytical Engine uses specialized relational algebra to keep the correct data flowing to each user geared to data-level security requirements without dragging down performance.

In Sisense, the Analytical Engine is seamlessly embedded into the query flow, starting with the dashboard and reports analytics experience through to the resulting query execution against the storage engine in play.

> "Our goal with Sense Analytical Engine was to deliver dramatic improvements in query performance, human-readable SQL for debugging and query validation, and full support for native database dialects for a wide range of data sources, including Redshift, Snowflake, BigQuery, Synapse, and others."

— Ziv Cohen
Engineering Director, Sisense

Analytical Engine breaks down query processing into a series of steps, starting with the logical query, JAQL. The query is then subject to multiple optimization steps, database-agnostic strategies, and database-specific SQL.



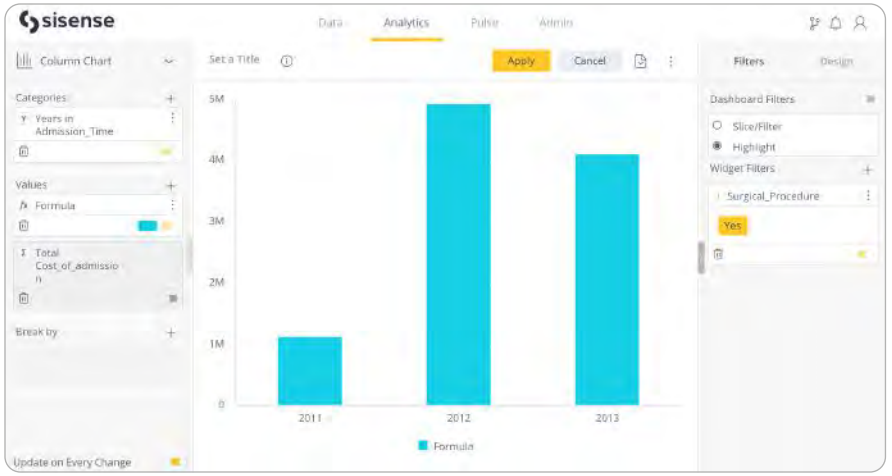
Semantic query definition using JAQL

In Sisense, query definition is enabled by JAQL (JSON query language), which began as an open-source project at Google, and provides a clear, declarative syntax. For example, a JAQL query clearly describes the measures and calculations in play at a business level without detailing the complexities of what the underlying data model ultimately requires for the query.

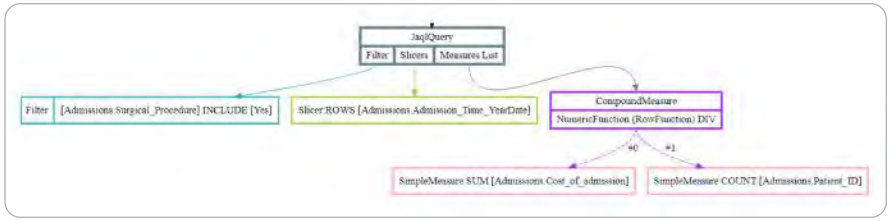
Representing the query in JAQL abstracts the consumer from the underlying data model, showing the semantics of the query instead.

In the example, the query clearly shows the admissions count measure in play, counting unique Patient_ID values filtered by whether Surgical_Procedure is Yes.

Sisense provides access to JAQL directly in the UX, enabling the business user and analyst to get a clear perspective of the query.



Example data visualization in Sisense Fusion



JAQL digraph underlying the visualization

Dimensional query formulation using DAQL

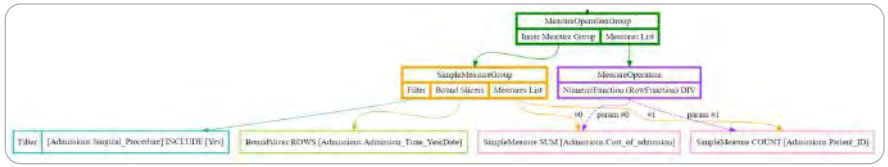
Sisense leverages a proprietary in-memory format for the first step of query optimization in the Analytical Engine, DAQL—dimensional query language, representing the logical query as a directed graph (digraph).

At the DAQL step, the Sisense Analytical Engine examines the JAQL query and maps it into the Sisense Data Model, taking cues from the context of the query in the reports and dashboards in play and generating the DAQL representation.

Initial DAQL digraph generated from JAQL before optimization

In the Sisense user experience, the DAQL graph can be inspected along with the original JAQL query.

At this step, the Sisense Analytical Engine applies dozens of intelligent optimizations, including patented optimizations unique to Sisense, in constructing the DAQL model.



Optimized DAQL digraph generated from JAQL using optimization

Some examples of dozens of optimizations are included at this step:

- Consistent path selection. Resulting DAQL always follows a consistent path, even when there are multiple "join paths," which only change if a user takes direct action to change the query.
- Optimized casting. Casting is applied minimally to optimize performance and only applied when it's a "must", such as when calculating an AVG or division of integers.
- Auto WHERE IN many to many join reduction (patent pending: 20220197950). Rather than using inner joins to apply filters, optimization will only use the WHERE IN to apply them. This improves performance on data sources where JOIN is expensive and also avoids creating 'many-to-many' JOINs. For example:

```
select *
    from fact_table
    join filter_table on key
```

becomes

```
select *
    from fact_table
    where key in filter_table
```

- Measure grouping. When measures are from the same table, the query scans that table once for calculating all measures rather than scanning the table for each measure. The Sisense Analytical Engine changes the generated code to calculate each measure only once.
- Data security optimization. Queries that use data security use push filtering to the innermost expression and are easily subsequently optimized by the partition mapper and query planner in Redshift, Snowflake, and BigQuery.
- Time series optimization. Implements BETWEEN filters for date compression rather than two separate filters.

Database-agnostic Relational Algebra Assembly

The Sisense Analytical Engine automatically builds a logical Relational Algebra from DAQL that incorporates the slate of optimizations as it's constructed.

Database-independent, Relational Algebra describes all the operations to fulfill the query, including required tables and fields, aggregations, filters, joins, types of joins, and table scans. The query is represented as a tree of relational operators.

```
LogicalSort(sort0=[$0], dir0=[ASC], fetch=[20000])
    LogicalProject(datetime_years_Admissions_Admission_Time_res=[$0],
    sum__Cost_of_admission__div__count__Patient_ID_res=[/(COALESCE(CAST($1):DOUBLE, 0),
    NULLIF($2, 0))], cond_exp_res=[CASE(IN($0, 2012-01-01 00:00:00), 1, 0)])
        LogicalAggregate(group=[{0}],
        sum__Cost_of_admission=[SUM($1)],count__Patient_ID=[COUNT(DISTINCT $2)])
            LogicalProject(datetime_years_Admissions_Admission_Time=[FLOOR($0,FLAG(
            YEAR))], Cost_of_admission=[$1], Patient_ID=[$2])
                LogicalProject(Admission_Time=[$5], Cost_of_admission=[$10],
                Patient_ID=[$1],Surgical_Procedure=[$8])
                    LogicalFilter(condition=[IN($8, 'Yes')])
                    JdbcTableScan(table=[[dbo, Admissions]])
```

Example Relational Algebra

The engine is pluggable, with Sisense continually adding additional algorithms, enabling companies to benefit from increasingly efficient queries with each release. Building an optimized DAQL model enables Sisense to render a highly database-agnostic query as Relational Algebra.

Database dialect-specific SQL

At the final step, the Sisense Analytical Engine converts the optimized database-agnostic query into ato database dialect-specific SQL using Apache Calcite.

Apache Calcite is an open-source dynamic data management framework and provides several key benefits:

- **Open-source –** Calcite is an open-source framework backed by the Apache Software Foundation that has been responsible for many of the most influential databases of the last decade.
- **Cross-system support –** The Calcite framework can run and optimize queries across multiple query processing systems and database backends.
- **Highly robust –** Wide adoption over many years has led to exhaustive testing of the platform.
- **Support for SQL and its extensions –** Calcite provides support for ANSI standard SQL and various SQL dialects and extensions.

In constructing the query, Apache Calcite examines Relational Algebra. It negotiates with the adapter to your specific database to find the most efficient way of accessing the data based on the supported dialect. Currently, Sisense Analytical Engine supports Snowflake, Redshift, BigQuery, Azure Synapse, and Databricks, and constantly adds support for additional databases.

```
SELECT
    "Admission_Time_YearDate_res",
    "sum__Cost_of_admission__div__count__Patient_ID_res"
FROM (
    SELECT
    "t1"."Admission_Time_YearDate" AS "Admission_Time_YearDate_res",
    COALESCE(CAST(SUM("t1"."Cost_of_admission") AS DOUBLE), 0) / (
    COUNT(DISTINCT
    "dTable_a083bb3f0LQAa5a96LQAa4088LQAab3d2LQAa8cd77393efd9"."value")
        AS "sum__Cost_of_admission__div__count__Patient_ID_res"
    FROM
        SELECT
        "aAdmissions"."aAdmissionXwAaTime_YearDate" AS "Admission_Time_YearDate",
        "aAdmissions"."aCostXwAaofXwAaadmission" AS "Cost_of_admission",
        "aAdmissions"."aPatientXwAaID" AS "Patient_ID"
        FROM
        "aSampleIAAaHealthcare"."aAdmissions"
        WHERE
        "aAdmissions"."aSurgicalXwAaProcedure" IN (
            SELECT
            "dTable_aAdmissions_aSurgicalXwAaProcedure"."key"
            FROM
            "aSampleIAAaHealthcare"."dTable_aAdmissions_aSurgicalXwAaProcedure"
            WHERE
            "dTable_aAdmissions_aSurgicalXwAaProcedure"."value" IN ('Yes')
        )
    ) AS "t1"
INNER JOIN
    "aSampleIAAaHealthcare"."dTable_a083bb3f0LQAa5a96LQAa4088LQAab3d2LQAa8cd77393efd9"
        ON "t1"."Patient_ID" =
            "dTable_a083bb3f0LQAa5a96LQAa4088LQAab3d2LQAa8cd77393efd9"."key"
GROUP BY
    "t1"."Admission_Time_YearDate"
) AS "it"
ORDER BY
    "it"."Admission_Time_YearDate_res"
LIMIT 20000
```

Final, human-readable SQL generated from the relational algebra

The benefit of this final translation step is clear. For example, in querying Google BigQuery it's best to use Google Standard SQL, which supports the broadest range of functionality, whereas Redshift performs better with its native, Postgres-style functions, data types, and query syntax. Optimizing for each dialect provides essential performance benefits.

Towards outcomes: Benchmarking performance

Analytical Engine delivers a significant speed-up from data to dashboard across a broad range of use cases, in many instances delivering up to 10X faster query performance and faster dashboard load times, all without the significant overhead and fragility of extensive query tuning. It delivers performance gains without the trade-offs associated with traditional business intelligence tools—such as sacrificing data security for speed or accepting poor performance from end-user-sourced queries.

Query length reduction

With the many optimizations applied during dimensional query formulation and assembly of the subsequent relational algebra, the Analytical Engine achieves significantly more efficient SQL. Across several benchmark queries covering domains from workforce analytics to sales analytics, testing showed between a 56% and 93% reduction in query length. Not only did this improve performance, but also provided substantial readability improvements for analytics developers who must verify and validate production queries.

Query performance improvement

Building a broad array of optimizations into the query, from reducing many to many joins to more efficient measure grouping, also provides significant performance gains, all utterly transparent to administrators and end users. Across three different benchmark queries, Analytical Engine increased performance from 64% to 87% without the need for manual optimization or ongoing tuning.

Reduced Query Length

56%–98%

Across several benchmark queries covering domains from workforce analytics to sales analytics, testing showed between a 56% and 93% reduction in query length

Increased Performance

23%

Across three different benchmark queries, Analytical Engine increased performance from 64% to 87% without the need for manual optimization or ongoing tuning

Conclusion

Visit us at www.sisense.com

Connect with us on

Putting analytics in the hands of everyone is now critical to businesses everywhere. Ensuring robust user engagement means analytics teams must deliver the best query performance that's both easy to scale, simple to understand, and gets the most out of their database investments.

The Sisense Analytical Engine provides the intelligence that organizations need to quickly turn business-intelligible queries into high-performance results. Organizations no longer have to compromise readability, maintainability, or data security to deliver business users' performance. It's a new era for analytics performance.

About Sisense

The smallest insights can drive large-scale business decisions.

Our goal at Sisense is to empower business leaders with a simple, streamlined process of getting the answers they need to get business done.

Sisense believes that in order to make better-informed business decisions, we need to remove the barriers between questions and answers where end users work. Because once there's a clear path to clear answers, business leaders can better understand their world and gain the necessary knowledge to take action in it.

Sisense embeds analytics seamlessly into any workflow. From retail to life sciences to manufacturing, our solutions power customer experiences by increasing user adoption and supporting smart, quick business moves. For product leaders, this means providing customers with uncomplicated tools for intuitive, insightful ways of working. In turn, key decision-makers get the answers they need in the tools they use, and the confidence to make data-driven decisions without waiting on analyst teams. With our analytics tools, IT and BI teams can help everyone at their company analyze, explore, and collaborate to uncover valuable insights.

We're not finding solutions for BI, we're helping people find answers. By streamlining the process of accessing insights, business leaders can gain the necessary knowledge to take critical action in a matter of minutes.

Ranked as the No. 1 Business Intelligence company in terms of customer success, Sisense has also been named one of the Forbes' Cloud 100, The World's Best Cloud Companies, six years in a row.