Table of contents

Introduction ......................................................................................................................................... 2
HP NonStop Systems Value Proposition .......................................................................................... 2

NonStop Kernel ................................................................................................................................... 3

HP NonStop SQL .................................................................................................................................. 3
   NonStop SQL/MP and NonStop SQL/MX ....................................................................................... 4
      NonStop SQL/MP Database ........................................................................................................ 4
      NonStop SQL/MX Database ........................................................................................................ 4

Why NonStop SQL? ............................................................................................................................ 5
   The Value Proposition of HP NonStop SQL .................................................................................... 5
      NonStop SQL Key Messages ........................................................................................................ 5

Unprecedented Scalability ................................................................................................................ 5
   Availability and Recoverability .................................................................................................... 6
      Availability features of NonStop Kernel ....................................................................................... 6
      Availability features of NonStop SQL ......................................................................................... 6

Disaster Recovery/Business Continuity ............................................................................................. 7

Mixed Workload Support ................................................................................................................ 8

Very Large Database (VLDB) Manageability .................................................................................. 8

Open Standards .................................................................................................................................. 8

NonStop SQL TCO ............................................................................................................................ 9

Distributed Database Architecture .................................................................................................... 9

Scalable Infrastructure Virtualization ................................................................................................... 10
   Data virtualization ......................................................................................................................... 10
   Transparent software virtualization ............................................................................................... 10

SQL Compiler .................................................................................................................................. 10

Executor Process ............................................................................................................................. 11

NonStop Data Access Manager (DAM) ............................................................................................ 12

SQL Connectivity Services .............................................................................................................. 14

ODBC driver .................................................................................................................................... 14
   Key features ................................................................................................................................... 14

JDBC driver ..................................................................................................................................... 14
   Key features ................................................................................................................................... 15

Unrivaled availability .......................................................................................................................... 15

Elimination of unplanned downtime ................................................................................................. 15

Fault-tolerant process pairing .......................................................................................................... 16

Elimination of planned downtime .................................................................................................... 16
Introduction

HP NonStop Structured Query Language (SQL) Software—the NonStop system’s database management system (DBMS)—delivers unequalled levels of data availability and performance for the most demanding applications with very large, multi-terabyte databases. It is the only standards-based enterprise DBMS specifically designed for clustered, shared-nothing system architecture. It is intrinsically easy to manage and delivers high performance in a cluster environment, providing unbeatable total cost of ownership (TCO) and Return on Investment (ROI). This is why NonStop SQL Software currently underpins most of the world’s stock exchanges, the majority of Point-of-Sale (POS) and Automatic Teller Machines (ATM) networks, and other large-scale business critical applications that cannot afford even a second of downtime.

This Whitepaper describes the NonStop SQL architecture and its value proposition.

Audience

This Whitepaper is intended for those who are interested in understanding the fundamentals of HP NonStop SQL.

HP NonStop Systems Value Proposition

HP Integrity NonStop servers offer a top-of-the-line enterprise computing solution. Designed from the ground-up for mission-critical environments, HP Integrity NonStop systems deliver unprecedented levels of application availability with fail-safe data integrity. Among the many distinct advantages of HP Integrity NonStop systems are the following:

- Continuous application availability—HP combines fault-tolerant hardware and software architecture with a comprehensive portfolio of innovative products and services to help ensure the customer’s NonStop infrastructure is available out-of-the-box, with no complex configuring providing a true definition of the term “24 / 7”.
- Linear scalability—The Nonstop environment allows customers to expand system resources with linear, unconstrained performance improvements. The NonStop system architecture has been demonstrated to scale up to petabytes of data per system and thousands of processors.
- Reliability, security, and data integrity—HP Integrity NonStop servers can protect companies’ business-critical data from multiple sources of risk.
- Real-time relational database offering massive linear scalability and parallelism (few hundred gigabytes to few hundred terabytes), uncompromised 24 / 7 database availability, absolute data integrity, and Very Large Database (VLDB) manageability. It provides complex and varied mixed workload support with demanding service level agreements (SLAs).
- Standardization—Based on Industry standard hardware components; HP Integrity NonStop servers can be managed by HP manageability software as part of its comprehensive enterprise management solution.
- Virtualization—HP Integrity NonStop servers offer virtualization capabilities enabling an environment that pools and optimizes all resources at the application level. With built-in clustering, automatic workload balancing and online management, NonStop customers can accommodate rapid growth seamlessly—without adding labor costs, compromising service levels or causing user disruption.
- Superior value—For servers in its class, HP Integrity NonStop servers deliver among the industry’s lowest total cost of ownership (TCO) with minimal downtime and ease of manageability.
- World-class support and services—HP global support and expert mission-critical consulting services are available around the clock and around the globe
NonStop Kernel

NonStop system is designed as a single entity in which hardware and software work together to provide fault tolerance, modular expandability, efficient networking, and other important functions. Programs can access a variety of services provided by the NonStop Kernel. These services make it possible to write applications that support the transaction processing functions and perform other tasks necessary for process execution, such as input/output and memory management. You access these services through one of the two environments provided by HP. HP provides two environments for developing and running applications on a NonStop system: the Guardian environment and the Open System Services (OSS) environment. Both environments provide commands, tools, and a set of procedure calls (the application programming interface, or API) that enable an application developer to access the services of the NonStop Kernel. Applications can be created and executed in either environment. The two environments are largely interoperable; that is, a process running in either environment can access the processes, files, tools, utilities, and API in the other environment.

The Guardian environment is HP’s proprietary interface to the operating system. The Guardian environment provides both a command-level interface and an API, which is a set of procedure calls an application can use to access the services of the operating system. These interfaces enable you to:

- Perform input and output on disk or tape
- Manage concurrent access to files so that multiple processes can access the same files without conflicting with one another
- Manage concurrent access to memory so that multiple processes can access the same items in memory without interfering with one another
- Send messages from one process to another and receive messages from other processes
- Write programs that execute as process pairs for fault tolerance
- Create and execute processes
- Manage memory such that when a process needs more memory, it can request a block of memory; when the process is finished with the memory, it can release it back to the operating system
- Debug programs at both a high (symbolic) level and a low (machine) level
- Compile and execute programs

Applications written for the Guardian environment fully support the NonStop principles. Open System Services (OSS) is HP’s environment for open systems development on NonStop systems. It provides an open alternative to the proprietary Guardian environment. OSS provides an interface to many of the services of the NonStop Kernel.

HP NonStop SQL

HP NonStop SQL is the mainstream database product for Integrity NonStop servers. It continues the HP NonStop SQL trend of aligning with industry standards, and it facilitates the porting of database applications to the NonStop platform. HP NonStop SQL Database is a clustered database system,
designed to leverage the shared-nothing, massively parallel processing (MPP) NonStop platform architecture.

**NonStop SQL/MP and NonStop SQL/MX**

NonStop SQL/MP gained the respect of customers and industry analysts as the world’s most scalable and available Relational Database Management System (RDBMS). NonStop SQL/MX, based on ANSI SQL standards compliance, enables even more popular applications, report writers, and tools to work with the world’s largest and most available, reliable databases.

**NonStop SQL/MP Database**

The HP NonStop SQL/MP relational database management system (RDBMS), is a truly parallel implementation of the structured query language (SQL) that is well suited to meet the demands of critical commercial applications, including online transaction processing (OLTP), batch operations, and query processing. NonStop SQL/MP software dramatically improves query processing by providing:

- Parallelism, or the ability to divide single SQL tasks into multiple subtasks that can be processed in parallel
- Scalability, or the ability to speed up queries for a database of constant size or to maintain the same response time for a growing database by adding processors
- Reliability, the combination of 24-hour-a-day, 365-day-a-year availability and full data integrity
- High performance, with advanced features such as total disk encapsulation, parallel execution, and hash joins

**NonStop SQL/MX Database**

NonStop SQL/MX—the latest version of NonStop SQL database—includes many features that enhance programmer productivity and facilitate the porting of applications to and from other database systems. NonStop SQL/MX delivers its advanced capabilities via industry-standard SQL syntax, Data Definition Language (DDL) manipulation, and other core characteristics. As part of the all-important business intelligence (BI) system that companies now depend on for competitive advantage, SQL/MX delivers the availability, scalability, performance and flexibility needed to get the job done right.

Taking advantage of HP heritage in parallel processing and high availability, NonStop SQL/MX Software provides:

- High performance by employing parallelism, a dataflow execution architecture, and advanced query optimization techniques to achieve outstanding response times in complex query environments such as Real-time Enterprise (RTE), Operational Data Store (ODS), and OLTP
- Scalability by supporting databases that can grow to multiple petabytes of data while retaining full online manageability with highly parallel and granular management operations performed at the table or index partition level
- Extensibility through an object-oriented architecture that can easily support new access methods
- Around-the-clock availability and data integrity through online management and industry-leading reliability
- Application portability through support of industry standards for database access
- Lower cost of ownership through industry-leading manageability features
- Interoperability and coexistence with NonStop SQL/MP Software
Why NonStop SQL?

The NonStop SQL Database is an inherently clustered database system, designed purposely to leverage a tightly-integrated, shared-nothing system architecture that scales linearly, is always available, offers unsurpassed reliability of fives and seven 9’s with lowest TCO in its class of servers.

NonStop SQL Software currently underpins most of the world’s stock exchanges, the majority of point of sale (POS) and ATM networks, and other large-scale business-critical applications that cannot afford even a second of downtime.

NonStop SQL is easy to manage and delivers high performance in a cluster environment, providing unbeatable total cost of ownership (TCO) and ROI.

The Value Proposition of HP NonStop SQL

NonStop SQL utilizes a massively parallel processing (MPP), shared-nothing architecture that allows it to scale up to 98.5 percent linearly from 2 to 4080 processors. Nonstop SQL is a fault tolerant hardware and software model with immediate and transparent take-over and unsurpassed data integrity. HP’s Nonstop Remote Database Facility (RDF) is a proven and high performance business continuity solution that can provide virtually instantaneous switchover of the database with zero lost transactions in the event the primary database encounters a disaster. RDF offers a wide range of deployment options. NonStop SQL supports industry leading management of massive multi-terabyte data stores through a “single clustered view” architecture that supports extensive online database manageability.

NonStop SQL Key Messages

- NonStop SQL is a key differentiator for NonStop solutions wherever a reliable, robust database is needed for complex OLTP and Decision Support workloads.
- NonStop SQL is the underlying database for complex real-time enterprise applications.
- NonStop SQL reduces complexity, delivers high performance, enables industry standard access, and high quality of service for superior value.
- NonStop SQL is tightly integrated into other system software and middleware on the NonStop platform for the highest reliability and performance.
- Applications and network managers have a single view of the clustered database even when distributed across the network. The database is designed to be managed and maintained (data reorganization, rebuilding indexes, disk replacement, and so on) online, providing 24 x 7 access to users.
- NonStop SQL is compliant with many industry standards (for example, ANSI SQL, Open Database Connectivity (ODBC) 3.5, and Java Database Connectivity (JDBC) 3.0) for data access, data manipulation, and catalog management.
- A full set of solutions, tools, and utilities—many from ISVs—for development and management make it a complete “ecosystem.”

Unprecedented Scalability

A NonStop SQL database is scalable—you can expand the size of the database simply by adding processors and disk volumes to the system. You do not have to change application code when you scale up the database. As the database and application grow, you can maintain the high
performance of the original, smaller database. Moreover, you can speed up the performance of a
database that is not growing—again, simply by adding processors and disk volumes.

The scalability of a NonStop SQL database is founded on the parallel hardware architecture. A
system contains from 2 to 16 processors, each with its own memory and disk storage, linked by a
pair of high-speed Servernet interconnects fabrics. Because the processors do not share memory or
disks, this is called a shared-nothing architecture.

NonStop SQL has been architected to take full advantage of the “shared-nothing” NonStop platform
architecture to provide high levels of linear scalability. The advantage of “shared-nothing” architecture
is that it minimizes contention among processors and other shared resources and thus achieves a very
high degree of scalability. Adding more processors and/or more disks enables the system to scale in
proportion to the power of the added components.

In order to scale, one also needs the capability to virtualize all processes running on the system and
the data across the entire cluster. Data is distributed using hash or range partitioning schemes across
all the disks in order to achieve that balance across the cluster. Also, queries are broken up into sub-
tasks each of which can be executed in parallel across the cluster. This provides the capability for
queries to scale as the NonStop system is scaled.

NonStop SQL offers “out of the box” 98.5% linear scalability. The computing power of a 16 logical
processor NonStop Blade server is equivalent to a 16-way node Unix cluster using the same
hardware. There are numerous NonStop customers who run NonStop SQL on 16 nodes (that are
called logical processors in NonStop terms).

Availability and Recoverability

For more than three decades, the HP NonStop platform has provided companies with continuous
availability. Fault tolerance was the primary design requirement of this integrated hardware and
software architecture, and no other company comes close to being able to claim such an extensive
experience base. Unburdened by lengthy recovery processes, application restarts, and offline
database maintenance needs, numerous long-time NonStop Server customers boast applications that
have never gone down for any reason.

NonStop servers were designed explicitly to provide absolute uptime; rapid, large-scale non-disruptive
growth; and uncompromising data integrity. The NonStop Server’s continuous availability is achieved
through the elimination of single points of hardware and software failure by utilizing redundant
hardware, patented process-pair technology, online database maintenance, and fault-tolerant
middleware, all based upon open standards such as SQL, UNIX, Java, and CORBA. Through a
combination of software process-pair technology, disk mirroring, and innovative online database
management, the database is available even if a processor, data access manager or a disk fails. And
NonStop SQL Software can be fully functional within minutes following a disaster such as a fire or an
earthquake.

Availability features of NonStop Kernel

The NonStop Kernel operating system uses process pairs to provide fault tolerance. A system process,
or an application process that requests database services from an application server process, can
have a backup process executing in another processor. If a fault occurs in the primary processor, the
backup process can take over without interrupting the application. Moreover, process pairs do not
require special programming. You can enhance the availability of the database by storing the data
for each disk volume on two identical disks. This feature, called disk mirroring, protects data from
hardware faults that could affect any individual disk. If one disk becomes inaccessible, the mirrored
disk continues to be available, and there is no impact on the application.

Availability features of NonStop SQL

NonStop SQL provides several features that make it possible to keep the database continuously
available to your application. Process pairs are especially valuable when applied to the data access
manager. You can configure the data access manager to have a backup process in another
processor. If a failure should occur in the CPU in which the primary data access manager resides, the
alternate data access manager can immediately take over the job of retrieving data so that the in-process query can complete successfully.

In addition, NonStop SQL ensures that application programs never have to fail when they access database objects that have been changed by the Database Administrator (DBA). The active data dictionary, which records all changes made to database objects and reflects the current status of the database, flags each application program that accesses a modified object. (Database objects include tables, indexes, views, programs, and collations.) NonStop SQL automatically recompiles an SQL query if the existing execution plan (the compiled, executable code) has been invalidated because of a modified database object.

In a distributed database, NonStop SQL can continue to process a query that accesses several table partitions even if a partition is unavailable. That is, the database remains available to the user even when a portion of the database is unavailable. The user can optionally skip unavailable data. If some of the requested data resides on the unavailable partition, the user still gets partial results. NonStop SQL also allows you to manage and reconfigure physical components of the database online; that is, users can continue full read and write access to the database while these management operations take place. For example, users can continue to access the database while the DBA performs backup and restore operations, loads data, and reorganizes files to maintain efficient access. Also, these operations can be executed at a lower priority than user queries so that they have a minimal impact on the performance of user applications.

You can perform the following tasks while the database is available for read and write access:

- Backing up the database
- Restoring a part of the database from backup
- Loading a portion of the database
- Adding a partition to increase database size
- The database is also available to user applications while you perform the following tasks:
  - Moving a portion of a table’s data to another disk (possibly a larger or faster disk) to balance I/O performance. This task is accomplished by splitting partitions or moving partition boundaries.
  - Creating an index on a table or reorganizing an existing table or index for optimal performance

At the end of these operations, however, the database is unavailable very briefly while database file labels are updated to reflect the changed structure. You can schedule the downtime so that it does not affect the execution of user queries.

**Disaster Recovery/Business Continuity**

Achieving elevated levels of availability or even the continuous application availability of NonStop servers—in a single site-system is still not enough to fully ensure business continuity for many types of critical applications. The sun literally never sets on an increasing number of companies, and they can no longer afford to have systems down while attempting to recover crucial applications. It is critical to geographically distribute application processing to protect it from a total site failure or regional disaster.

HP provides software to protect against a disaster affecting an entire site. NonStop SQL works with the Remote Duplicate Database Facility (RDF) product, which lets you maintain a duplicate database on a separate system at a remote location. As users modify the primary database, RDF replicates the changes on the duplicate (backup) database. You can configure RDF so that multiple remote systems
maintain backup databases for one another, providing data replication. Each system can then support primary application activity (such as OLTP or query processing) as well as providing for disaster recovery for another system. Through the use of HP NonStop Remote Database Facility (RDF) software, a target (that is, remotely located) NonStop server can take over from a source server in a time frame measured in seconds.

NonStop RDF software has nearly a two decade track record of multi-site business continuity, with over 800 NonStop software licenses across more than 350 customers. Working together with NonStop RDF Software, NonStop AutoTMF and NonStop AutoSYNC, these products provide virtually instantaneous switchover of the database with zero lost transactions (ZLT) in the event the primary database encounters a disaster. No other database vendor provides these levels of assured protection for mission-critical transactions and data.

Mixed Workload Support

NonStop customers have shown that very large real-time enterprise environments require the simultaneous support of many different users and applications. HP calls this “mixed workload.” For example, a mix might include simultaneous support for:

- Large scale and real-time ETL requirements (extract, transformation, and loading)
- Large scale OLTP or operational access
- Online subscribers of the data such as real-time event analysis, real-time fraud detection, real-time reporting, real-time data mining, and so on
- Batch subscribers such as data mart extracts, enterprise application integration data feeds, and so on
- Complex query support such as OLAP (Online Analytical Processing) and complex reporting
- Database maintenance. As the database becomes enterprise critical or the core of e-commerce, it requires availability around the clock (24x365). This implies maintenance operations need to run concurrent with transaction processing
- HP NonStop SQL utilizes a simple and yet extremely powerful priority based model to control mixed workload execution. Priorities can be assigned to individual workloads (that is, queries and transactions) based on the user SLAs. This mechanism is unique to HP NonStop SQL and ensures unequaled mixed workload support.

Very Large Database (VLDB) Manageability

One of the biggest challenges of creating a Real Time Enterprise (RTE) solution is that the data store at the core of the solution is extremely large, typically many terabytes in size. VLDB (very large database) manageability is the ability to monitor, configure, and initiate maintenance command and control over very large database objects online (no interruption of service including full read, write, and update during operation), globally (no matter where the object is running in one or multiple distributed servers), and with fewer resources (automated operations and single system image).

NonStop SQL supports industry leading management of massive multi-terabyte data stores through a “single view” architecture that supports extensive online database manageability and highly parallel and granular management operations performed at the table or index partition level. These partition level operations support online and parallel execution providing dramatic reductions in the amount of time required to complete the operation.

Open Standards

NonStop SQL is based on open standards. Open standards support database connectivity that allow more applications, solutions and third-party products to work with the HP SQL database and platform. The ANSI/ISO SQL standards:

- NonStop SQL/MP – ANSI’89 standards
- NonStop SQL/MX – ANSI’03 standards
Embedded SQL support in C, C++, and COBOL

Stored Procedures in Java support

Database connectivity standards:

- Open Database Connectivity (ODBC) and Java Database Connectivity (JDBC), play an important role in portability by providing an open, standards-based connection vehicle to the database applications and query and reporting products.

**NonStop SQL TCO**

NonStop Servers have been recognized for many years as the leader in Total Cost of Ownership (TCO) for business critical applications. TCO includes not only the initial investment costs (hardware and software purchase costs) associated with a DBMS but also the ongoing costs of ownership. Included in TCO costs are ongoing costs such as licensing/maintenance fees and administrative and training costs for running the database. True TCO also takes into consideration the true costs associated with downtime (lost revenue, lost productivity, impact on customer satisfaction).

**Distributed Database Architecture**

NonStop SQL is architected for continuous availability, extreme scalability, high performance for complex workloads, and mixed workload capability to meet the needs of mission critical real-time workloads.

NonStop SQL is based on an underlying MPP (Massively Parallel Processing) platform architecture. The shared-nothing aspect of this architecture means that there is no single resource, such as disk, memory, etc. that is shared amongst the processors. This avoids any bottleneck on a single resource, thereby providing the capability to scale linearly to a large number of processors in a cluster with each processor being a parallel unit of work. NonStop SQL has been architected to take full advantage of this platform architecture to provide high levels of scalability.

In order to scale, one also needs the capability to virtualize all processes running on the system and the data across the entire cluster. Data is distributed across all the disks in order to achieve that balance across the cluster. Also, queries are broken up into sub-tasks each of which can be executed in parallel across the cluster.

One of the important differentiating factors of NonStop SQL is its ability to handle OLTP and mixed workloads concurrently. This is because our OS and database stack is designed such that priorities drive the servicing of a workload. The priority of a query or transaction governs what and how quickly processor and I/O will be allocated to that workload. So if the OLTP workload is running at a higher priority, then long-running analytical queries will not impact this workload whatsoever.

Another important aspect is the highest level of data integrity that is inherently part of NonStop. Transaction management performance has been optimized and in fact speeds up updates since cache writes do not have to be flushed to disk right away as they are backed up by the audit that is written to the logs very efficiently. The database dump and audit logging capabilities provide full recoverability to a point in time if there is dual hardware failure or a user error such as a table being dropped or a bad update or load needing the database to be recovered to a point in time before that operation was performed.

NonStop servers provide unrivalled availability not only via hardware and OS fault tolerance that tolerates any single point of failure but also from a database perspective where all operations are completely online. The database does not have to be brought down for table reorganization, index
creation, or any other routine database maintenance operation. When we use the terminology “online” it is not only for allowing reads against the database but also updates. These operations can be performed online at the appropriate priority so that the operation is either completed fast or paced so as not to impact other workloads. In certain cases, such as index creation, the operation is performed in the background and at the end of the operation it applies all updates that it reads from the audit logs. Once it is caught up with the updates it brings the offline object online, such as making the index available for use, thereby making the whole operation seamless while providing full read and write capability throughout the process.

Scalable Infrastructure Virtualization

A processor is a unit of parallel work. NonStop platform is scalable from 2 to 4080 processors. Both data and software is virtualized across the entire cluster.

Data virtualization

NonStop SQL allows you to partition the data in a table across many disk volumes. Within each partition data is clustered currently based on the primary key. If specified, the hash key has to be the same or a subset of the clustering key or primary key columns. This clustering of data, if designed appropriately, can be leveraged for very fast range access to data within a partition for a specific query. In fact, it can be leveraged even more effectively with the use of patented technology called Multi-Dimensional Access Method (MDAM) which allows effective pruning of sets of data even if values for leading or intermediate key columns are not provided, range predicates are provided on the leading columns with equality predicates on non-leading columns, or a list of values is provided for a key column.

You can configure your hardware system so that the disk volumes (and, thus, the table partitions) are managed by different processors within a system or different processors across a distributed network. Partitioning the database allows NonStop SQL to take full advantage of the parallel processing.

NonStop SQL provides location independence. You can store data where it is used most frequently, and your applications can retrieve and change the data regardless of where it is located. Accessing distributed data is as simple as accessing local data: the applications provide the names of the tables that contain the data. NonStop SQL determines the location of the data as well as the best way to retrieve it.

NonStop SQL maintains a distributed data dictionary that describes the objects in a distributed database. If a table or index is partitioned among several nodes, NonStop SQL duplicates the data descriptions in catalogs residing on each node, so that each partition is described on its local system. Each catalog also identifies the location of all other partitions. The data dictionary thus provides a unified logical database schema that gives applications complete access to the distributed data. In addition, NonStop SQL provides local autonomy. In a distributed database, all NonStop SQL systems in the network cooperate with one another, yet each system can manage and access its own local data independently of the other systems.

Transparent software virtualization

NonStop SQL is fully integrated with the NonStop Kernel operating system so that it can take advantage of the distributed processing made possible by the hardware architecture. Through the NonStop Kernel message system, a NonStop SQL process accesses a remote resource simply by qualifying the resource with a node name. NonStop SQL is also fully integrated with other software to ensure that high performance as well as the integrity and availability of the data is maintained. Integration with other parallel products is especially important in a large, distributed system in which many NonStop SQL processes execute in parallel.

SQL Compiler

Before a query can be executed, it needs to be compiled by the SQL Compiler. The compiler:

- Parses the query and builds a query tree checking the syntax in the process
Does binding: accesses the catalog information and resolves and understands all the object references in the query such as the table, columns, and associated objects of the table such as indexes, stored procedures, materialized views, histogram statistics, etc. that may be relevant to the query and the generation of a plan for it. It also decomposes views at this stage.

Normalizes the query tree transforming certain parts of the query to more efficient constructs such as transforming sub-queries into joins.

Finds the most optimal plan based on histogram statistics information that it collects during the binding phase in the optimization phase. The optimizer is a rules driven cost based top-down optimizer based on Cascades optimization technology and Large Scope rules that try and identify join patterns such as star joins to quickly generate the optimal plan for a complex query.

Generates the structures for this plan to be passed to the execution engine that can then execute this plan.

The plan generated for the query is cached. If a query is seen again that can leverage the same cached plan then it does so avoiding the compilation of the plan again. This caching mechanism is very sophisticated.

Executor Process

Associated with every query there is a Master Executor that essentially is the conductor that executes the query and collects and collates the results of the query and passes them back to the client application.

Multiple NonStop SQL executor processes (ESPs) can execute in parallel in separate processors. The SQL executors can process multiple small queries or individual operators of a query in parallel. A query can be broken up into a scan operator that scans the table; join operators that may perform various different types of joins such as nested, merge, or hash join between two tables, based on the type it considers the most efficient for that join operation; unions; partial aggregation to reduce message traffic early and to keep joins smaller before a final aggregation is done; full aggregations, sorts, update operators, etc. For a large query, an SQL master executor process can create executor server processes that divide up the query into smaller tasks and process the tasks in parallel. Loading, administrative, and management tasks are also executed in parallel.

The architecture of the execution engine is data-flow architecture. That is, data flows down from parent operators to child operators and from child operators to parent operators in queues. So as a scan is done scanning a buffer full of rows it puts it into an up queue for say an aggregate operator. The aggregate operator, as it is done with its aggregations will flow the results of its aggregations into up queues say to a join operator. If it is say a nested join then the join operator will flow the rows down to the scan operator to find matching rows in the second table with the resulting matching rows being returned by the scan to the join via an up queue. The join then returns the results to its parent which may be the root operator. Each operator is scheduler-driven and is performing its task independent of the other operators as it gets scheduled. So, unless there is a blocking operator, such as a sort or certain aggregations, the data is continuously flowing down and up the query operator tree. No one operation is being fully executed before another one begins (except for the blocking operators as mentioned).

The other key aspect of this architecture is that the data flows continuously from one operator to another and does not land on disk in most cases. In most cases – depending on the query workload for the customer – almost 80% to 90% or more of the queries can be processed with the data flowing through memory and never landing on disk. Since each operator is getting queues full of data at a time, and is getting a small subset of the data since the query is parallelized, most of the time it can
do the operation by keeping the data in memory. However, if large amounts of data qualify and are
being sorted for an order by or are part of a hybrid hash join or group by where the operator does
not have enough memory to process all the data, then data does overflow to disk.

Finally, NonStop SQL leverages three types of parallelism. One is partitioned parallelism that is
influenced by the fact that the data is partitioned and therefore separate parallel operators will be
processing data from each of those partitions in parallel. Then there is pipeline parallelism. The
queues between operators could be within the same process or in separate processes depending on
whether the optimizer decides to put a parent and child operator in the same or different process. In
the same process they are queues in memory while across processes they are in the form of inter-
process messages. Because of the pipelining often the parent and child operators are working in
parallel. And then finally there is independent, or operator parallelism, such as where the two scans
for the two tables are independently scanning the tables in parallel.

These features make it possible for a NonStop system to perform massively parallel processing against
a very large database. Moreover, these features allow you to scale a database to a larger and larger
size as your application requirements grow, without affecting performance.

NonStop Data Access Manager (DAM)

The NonStop SQL database management system (DBMS) achieves high performance because it
operates at the lowest levels of the NonStop server architecture. Instead of operating as a software
layer above the operating system, NonStop SQL is integrated with NonStop Kernel system processes
such as the file system, message system, and data access manager (DAM previously known as DP2
disk process).

In a traditional DBMS, the DBMS must perform most of the query processing, communicating with low-
level system processes only to retrieve the data. With NonStop SQL, the low-level system processes
themselves perform most of the query processing. The NonStop SQL architecture considerably
shortens the path length to the data residing on disk, significantly enhancing performance.

The DAM evaluates SQL requests made on a single table or table partition. Because the data access
manager is fully responsible for data access to a single disk volume, it can be said that the access is
capsulated—that is, no other process can access the data and thus subvert the DBMS.

A data access manager can retrieve data from the table based on the selection criteria in the query; it
can also delete and update data. The data access manager executes SQL requests as close as
possible to the data, eliminating message traffic to and from a higher-level DBMS layer, as happens
in most DBMS systems. To optimize performance, NonStop SQL is distributed within the NonStop
server so multiple processes can execute separate SQL requests simultaneously or divide a large
request into separate tasks and process the tasks in parallel. NonStop SQL uses the message system
to pass requests to the data access managers and return data to the processors in which the requests
originated.

The SQL executor component of DAM does projection – i.e. it extracts only the columns needed by
the query. Selection, or the application of predicates against the data being scanned, is pushed
down to DAM as well if it possibly can. The optimizer tries to push predicate evaluations as low in
the execution tree as possible so that fewer rows and messages flow up the execution tree. As
mentioned earlier, aggregations needing little memory can also be done in DAM.

Since a disk volume is mirrored, DAM manages both the primary and the mirror. It writes data in
parallel to both the primary and mirror drives. When scanning, it can scan from either the primary or
the mirror depending on the drive that has a head in proximity to the data being accessed.

When the desired data resides on a remote system in a distributed network, the data access
managers on the remote system filter the data, returning only what you requested. This design is
especially beneficial for network performance because it reduces the size and number of messages
sent across the communication lines.
NonStop SQL organizes special memory pages, called cache, where data can be stored temporarily in memory. Data buffers retrieved by the data access manager are read into cache so that the most frequently used data is in memory. This strategy reduces the frequency with which data must be retrieved from disk, thereby improving performance for both transactional (random) and sequential access to data. When database records need to be read sequentially to satisfy an SQL request, the data access manager can buffer the records, sending blocks of data rather than a single record at a time. A valuable function of the data access manager is that it can filter the data as soon as it is retrieved from disk and then send buffered blocks of data. Both types of buffering reduce the messages and data sent back to the SQL file system, improving performance for sequential access.

DAM does priority management which is at the heart of the mixed workload capabilities. When a query is launched at a certain priority that priority determines how it gets serviced by both the processor for processor resources as well as by DAM for I/O resources. DAM always works on higher priority workloads before it will work on lower priority ones. As it is working on a query it will pause every so often to check if any higher priority workload is requesting services from it. If there is, it will switch over to working on that workload. If not, it will continue working on the workload it is processing. However, if a high priority workload is using up a lot of resources DAM will automatically lower its priority so that lower priority workload does not get starved out.

DAM is also a process pair, something that will be discussed later in availability. That is, it checkpoints to a backup process on another processor. If there is a processor failure DAM will switch over to its backup process and will continue to manage the disk volumes it owns.

DAM however does a lot more since it is managing the disk volume and the data on it. It does row locking in order to provide consistency and data integrity based on the various isolation levels used, such as read uncommitted, read committed, or repeatable access against updates that may be taking place. Read uncommitted does not lock a row nor does it wait on locked rows. It reads through locks or does dirty reads. Read committed does not lock a row but does wait on locked rows to be released before it can read further. A repeatable read locks all the rows it reads, releasing them only at the end of the transaction, to prevent any other transaction from updating any rows being accessed by it. NonStop SQL also supports SKIP CONFLICT where if a row is locked it will be just skipped instead of the read committed waiting on it. This is mostly used when a database-level Publish Subscribe feature is used for de-queuing rows that are not being de-queued by other subscribers. Deadlocks are avoided via default timeouts i.e. one of the transactions waiting on the lock will timeout thereby breaking the deadlock. Since each of the DAMs manages locks for a single disk volume, no lock coordinator is needed – i.e. NonStop SQL has distributed lock management. NonStop SQL extends the isolation control from the ANSI transaction level controls to SQL statement and even to table level controls. So in a single statement, the user can read one table with a read committed access and another table being joined to it, say a dimension table, with read uncommitted access.

Related to locking is transaction management. DAM writes all updates (before and after images) to the transaction audit trail, or audit log, even before it writes the data to disk. So database changes for a committed transaction are always in the audit log and can be used to recover the database to the point of that commit, even if a failure would cause the data not to be written to the data disk itself. The updates are written out to their respective disk partitions from cache whenever DAM gets the time to do so. The entire database is dumped, or backed up, online (i.e. while the database is available for reads and updates). With the audit logs, this provides full recoverability of the data to a point in time if say a table was dropped by mistake or an update operation or load was done erroneously and the database needs to be recovered to a point just before that operation. This audit can also be used for a Removal Media for Disaster Recovery (RMDR) solution, where in the case of a disaster the
entire database can be restored to another NonStop server to a consistent state as of the last audit log.

With the locking and transaction support, NonStop SQL provides very sophisticated transactional controls for total data integrity and recoverability.

**SQL Connectivity Services**

HP provides a broad range of connectivity options to the NonStop SQL database platform.

The ANSI SQL-compliant NonStop SQL database can be accessed using Open Database Connectivity (ODBC) 3.0 and Java Database Connectivity (JDBC) Type 4 interfaces from Microsoft Windows, Linux, HP-UX, IBM AIX, and Sun Solaris-based platforms and an on-platform JDBC Type 2 native driver and OSS ODBC/MX driver.

### ODBC driver

Open Database Connectivity is a common framework for accessing a database platform. An ODBC driver is installed on the client platform that supports the database platform. Developers use an open standard application program interface (API) to access the NonStop SQL database. The API specifies standardized representation of data types, and a library of function calls to execute a SQL statement to request and retrieve results as a single row or as result sets. The client application need not know the native database interfaces. The ODBC driver handles the request and converts it into one that NonStop SQL understands and executes.

HP NonStop has made an on-platform native ODBC driver as well as an off-platform driver usable by all Windows-based clients.

The API is flexible and enables:
- Construction of SQL statements at compile or runtime
- Access to multiple instances of a database
- Sending and receiving data in a format convenient to the client application while relying on the driver to do the conversion between the database and application platform

ODBC driver is released as a DLL and users can link their application with this driver.

**Key features**

- Out-of-the-box ODBC 3.0 standards compliant
- Available for many data sources
- Supports multiple, concurrent applications through a single driver instance
- Provides both ANSI and Unicode access to databases
- Supports 32-bit and 64-bit (R3.0 onwards) connectivity
- Supports failover detection and automatic recovery. Single failures affect, at most, a single connection. All other application connections remain unaffected by the failure of one application connection
- Utilities to install, remove, and query installed drivers
- ODBC Administrator
- ODBC Test Utility

### JDBC driver

The Java Database Connectivity API is a set of classes allowing a vendor neutral access to a database from within a Java application.

NonStop provides two JDBC drivers:
• JDBC Type 2 driver consists of Java wrappers to the SQL Command Line Interface (CLI). It offers better performance but is local to the NonStop platform. It translates JDBC function calls to native SQL calls.

• JDBC Type 4 driver is a pure self-contained Java implementation. The driver is loaded in the same Java Virtual Machine (JVM) as the client and calls database APIs.

These drivers are JDBC 3.0 compliant and run on all platforms supporting Java Development Kit (JDK) 1.4 or later. The drivers also support JDBC API for connection and statement pooling and datasources.

**Key features**

• Out-of-the-box JDBC connectivity to NonStop SQL database
• Available for many supported data sources
• JDBC 3.0 standards compliance
• Runs on all platforms that support Java Development Kit (JDK) 1.4.1 or later
• Supports Dynamic SQL, Java Stored Procedures (SPJs), metadata API, Unicode, I18, row sets, result sets, and Binary Large Objects (BLOB)/ Character Large Objects (CLOB) data types
• Supports XA Resource Manager that enables NonStop SQL to participate in distributed transaction environments
• Supports 32-bit and 64-bit (R3.0 onwards) connectivity
• Supports thread-safe non-blocking behavior
• Optimized for superior performance with extensions to JDBC like connection pooling, statement caching, static module file caching, row sets, and result sets. NonStop SQL Connectivity Services (MXCS) TCP protocol has been tuned for very fast data access.
• Single failures affect, at most, a single connection. All other application connections remain unaffected by the failure of one application connection.

In summary, NonStop Enterprise Connectivity provides a complete set of standards-based drivers and APIs for connectivity applications and platforms with NonStop SQL database, supporting a variety of languages and environments.

**Unrivaled availability**

**Elimination of unplanned downtime**

The unrivalled availability of NonStop servers addresses availability for both unplanned downtime and planned downtime.

The hardware itself is reliable and resilient to failure. NonStop servers use built-in hardware redundancy and fault tolerance and can handle any single point of failure, and often many dual failures.

First, there is the RAID1 disk failure protection provided by the fact that if the primary disk fails, there is always access available to the backup disk drive.

In case of a dual disk failure, where the primary and mirror disks fail at the same time, however unlikely that is, the NonStop SQL database dump and audit logging capabilities provide full
recoverability up to the last committed transaction. There is downtime involved in replacing the bad primary disk and performing the recovery. The mirror can then be replaced, and is refreshed while the primary drive is fully available for read, and write, operations. After the mirror is refreshed, it also is available for reads and writes.

Then there is the controller failure protection provided by the dual controllers that connect to the drives. If the primary controller fails, the backup controller is used to access the data.

And finally, there is the ServerNet fabric that also has built in redundancy so that a failure to one part of the fabric leads to an automatically switching over to the backup fabric.

**Fault-Tolerant process pairing**

As discussed above, there is hardware redundancy with the mirrored drives, redundant I/O channels, and redundant ServerNet fabric where there is a path to the data regardless of any single hardware failure. Process pairing is used by DAM where a primary process is running on one processor managing a disk volume with a backup process running on another processor. The primary process is constantly check-pointing to the backup process. If the primary processor fails DAM processing switches instantaneously to the backup process with all access to the disk volumes now being managed and available by the backup DAM processes. So it is instantaneous takeover versus failover. That is, failure does not interrupt the availability of the database. There is no restart of the database or recovery operations required to make the database available again. The data base is always available. DAM is a key process in providing the continuous availability of the database and therefore leverages process pair technology.

Another process associated with transaction management ($TMP) is also a process pair since data integrity is so key to NonStop. The transaction management system has to always be there to coordinate and ensure complete data integrity. There are many other such process pairs on the NonStop system to provide instantaneous access to a resource if there is a failure of the primary process for any reason what-so-ever.

There are many other important processes in NonStop as well, but having them set up as persistent processes is sufficient. A persistent process is where a daemon knows about this process and if it ever goes down due to a software or hardware failure the daemon will start the process again. A lot of thought has gone into making the system resilient to failure to provide the highest levels of continuous availability.

Process pairs are used over persistent processes only when there is a potential for loss of data or data integrity or when state needs to be maintained that one cannot afford to lose.

**Elimination of planned downtime**

Planned downtime is eliminated as well by all workloads being able to run concurrently with full read and write access to the database, during loads, reorgs, or index creation.