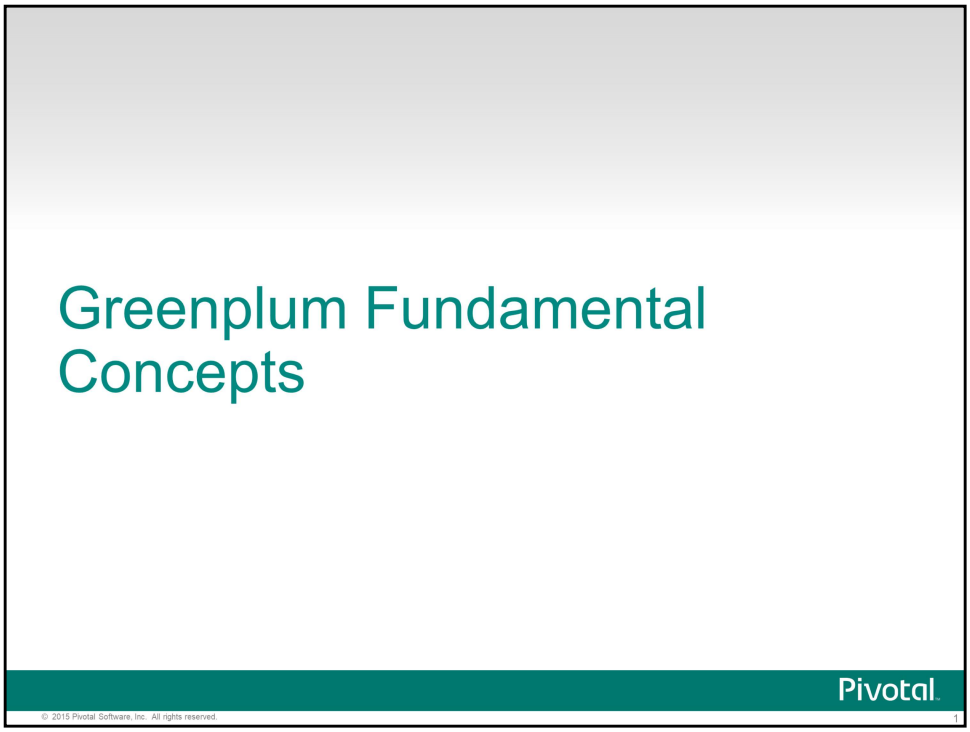


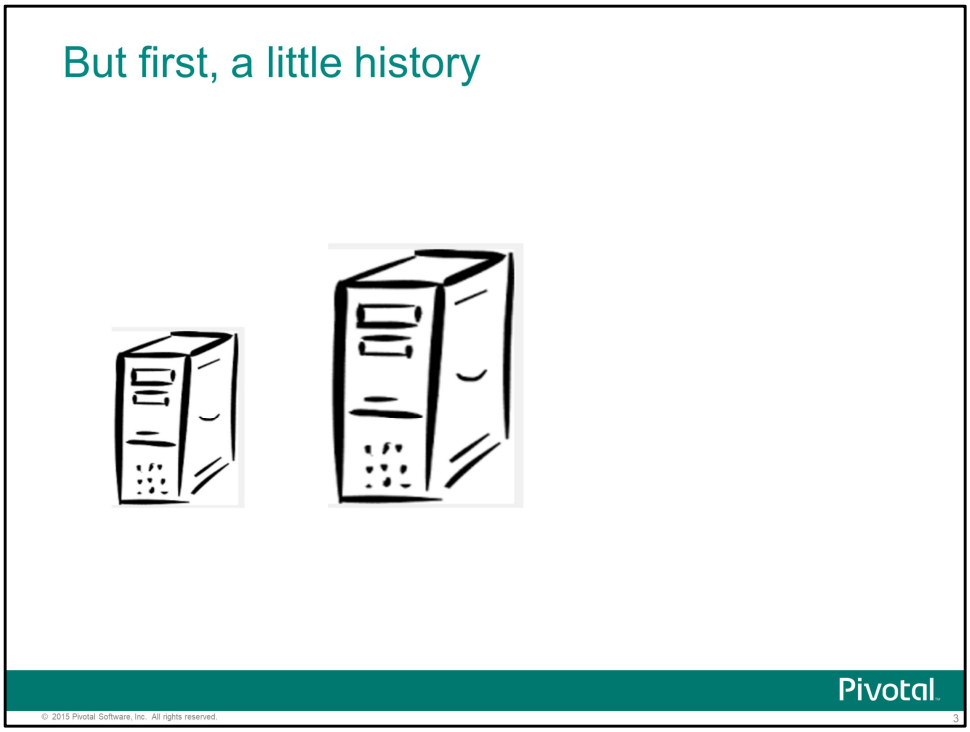

Hello. This is Marshall Presser from Pivotal Engineering introducing the Pivotal Greenplum Database. For the next few minutes, we'll explore the GPDB architecture and explain why the GPDB is proficient in doing Big Data Analytics. Understanding the architecture will help you make effective use of the GPDB. If you're listening in here, you are likely to be part of an organization that has a huge amount of data to load and analyze in order to make effective business decisions. We're going to help you do that. <CLICK?

## But first, a little history

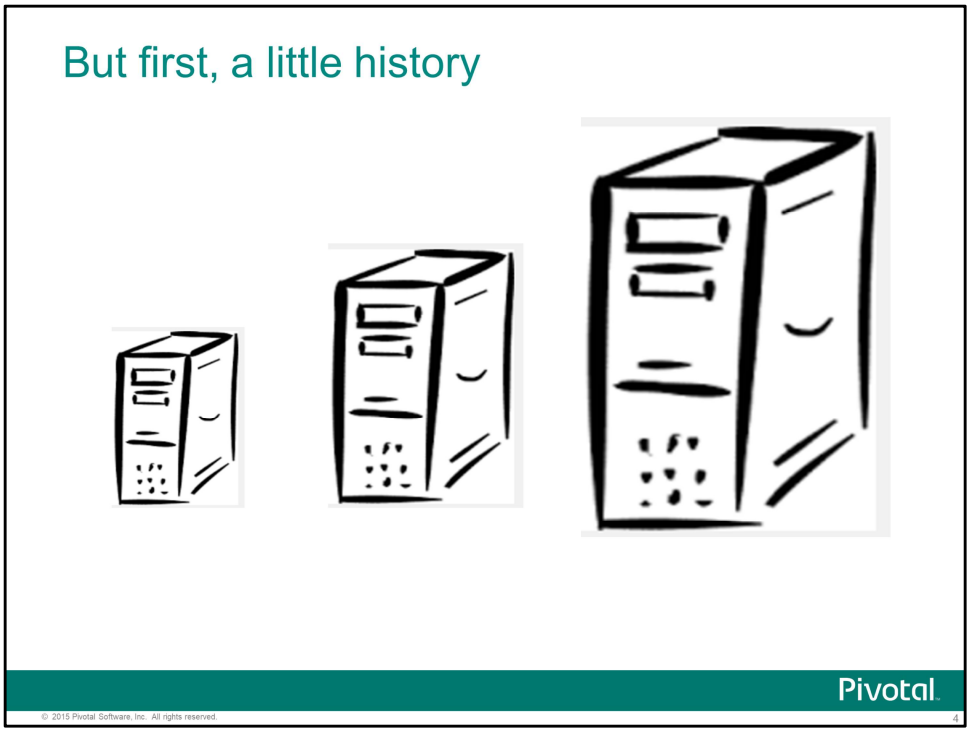
Pivotal




When people first moved away from expensive mainframe computers, they did their data analytics in Symmetric MultiProcessors. They would start with a small box <CLICK> but would soon outgrow it and then need to move to a larger one <CLICK> , and when they outgrew it, move to a yet larger one. <CLICK> Costs became prohibitive and performance began to suffer. So, people in the data world became to do what people in the scientific computing world had done: scaling out rather than scaling up. <CLICK>




When people first moved away from expensive mainframe computers, they did their data analytics in Symmetric MultiProcessors. They would start with a small box <CLICK> but would soon outgrow it and then need to move to a larger one <CLICK> , and when they outgrew it, move to a yet larger one. <CLICK> Costs became prohibitive and performance began to suffer. So, people in the data world became to do what people in the scientific computing world had done: scaling out rather than scaling up. <CLICK>




When people first moved away from expensive mainframe computers, they did their data analytics in Symmetric MultiProcessors. They would start with a small box <CLICK> but would soon outgrow it and then need to move to a larger one <CLICK> , and when they outgrew it, move to a yet larger one. <CLICK> Costs became prohibitive and performance began to suffer. So, people in the data world came to do what people in the scientific computing world had done: scaling out rather than scaling up. <CLICK>

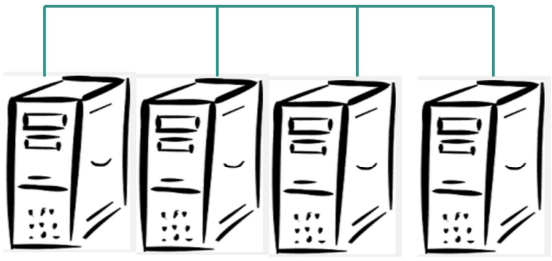
## Massively Parallel Processing (MPP)
## Scaling out, not up.

Pivotal




But in an MPP, scale out architecture, we increase computing power and storage by starting small <CLICK> In most MPP architectures today, the nodes are usually small Linux servers with enough CPU, RAM, and I/O power for the needs at hand. When we run out of any of these resources , we add another node or more . <CLICK> Since each node has its own memory, OS, CPU and storage with its own set of disks, this architecture is known as a Shared Nothing architecture.

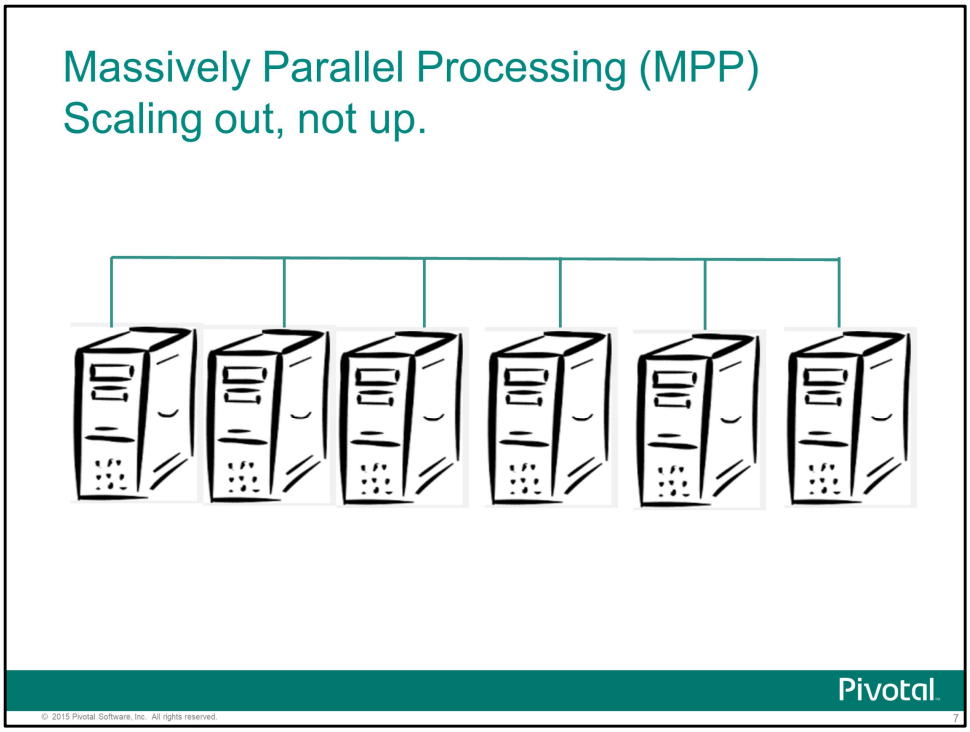
## Massively Parallel Processing (MPP)
## Scaling out, not up.



Pivotal

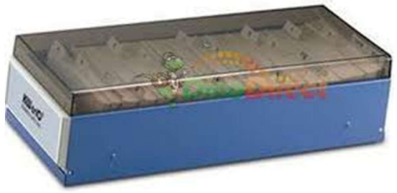



But in an MPP, scale out architecture, we increase computing power and storage by starting small <CLICK> In most MPP architectures today, the nodes are usually small Linux servers with enough CPU, RAM, and I/O power for the needs at hand. When we run out of any of these resources , we add another node or more . <CLICK> Since each node has its own memory, OS, CPU and storage with its own set of disks, this architecture is known as a Shared Nothing architecture.




But in an MPP, scale out architecture, we increase computing power and storage by starting small <CLICK> In most MPP architectures today, the nodes are usually small Linux servers with enough CPU, RAM, and I/O power for the needs at hand. When we run out of any of these resources , we add another node or more . <CLICK> Since each node has its own memory, OS, CPU and storage with its own set of disks, this architecture is known as a Shared Nothing architecture.

## A brief analogy

Pivotal



8

A colleague has a large collection of business cards. He exchanges business cards with everyone he meets. He has collected 10,000 business cards.

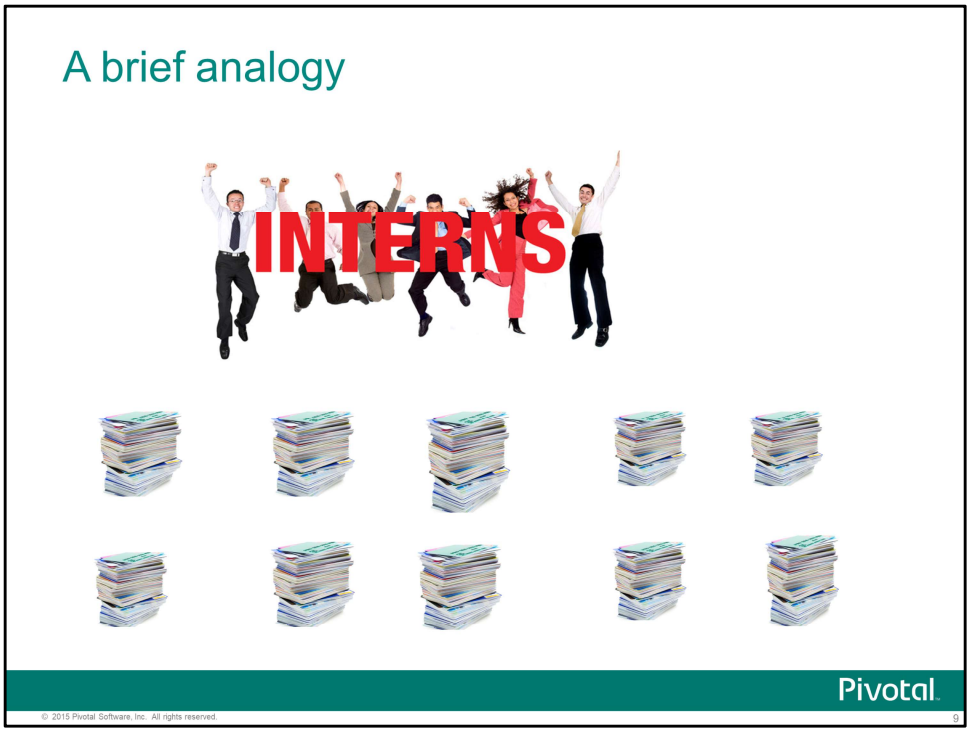
Being no fool, he keeps them sorted in alphabetical order: last name, first name. When he needs to look up someone's contact info and he knows the name, it's a no-brainer. He goes to

roughly the write place in the alphabet and scans a few cards, taking a few seconds.

Occasionally he needs different information  He’s

visiting Acme Widget and want to find all his contacts who work for them. This requires going through all his cards. Now he can scan through cards at a pretty rapid rate, say 100/minute. For this

current card stack, this takes 10,000/100=100 minutes. This is too long and he came to me for advice.




"Don't you have a lot of interns working for you? Why not enlist the help of 10 of them, divide the cards equally among the 10, and have each intern scan the cards in parallel?"
10,000 cards/10

interns=1000 cards/intern. Assuming the interns can also scan cards at 100/minute, they can get the job done in 10 minutes, a 10X speedup over a single user. If we had used 50 interns, the

job could be finished in 2 minutes, a 50X speedup. We get linear scalability from this strategy.

Our parallel solution has the advantage of faster lookups. And

my friend can use a similar parallel strategy for loading news sets of cards he gets at the next Strata conference.

Notice that the process is the same

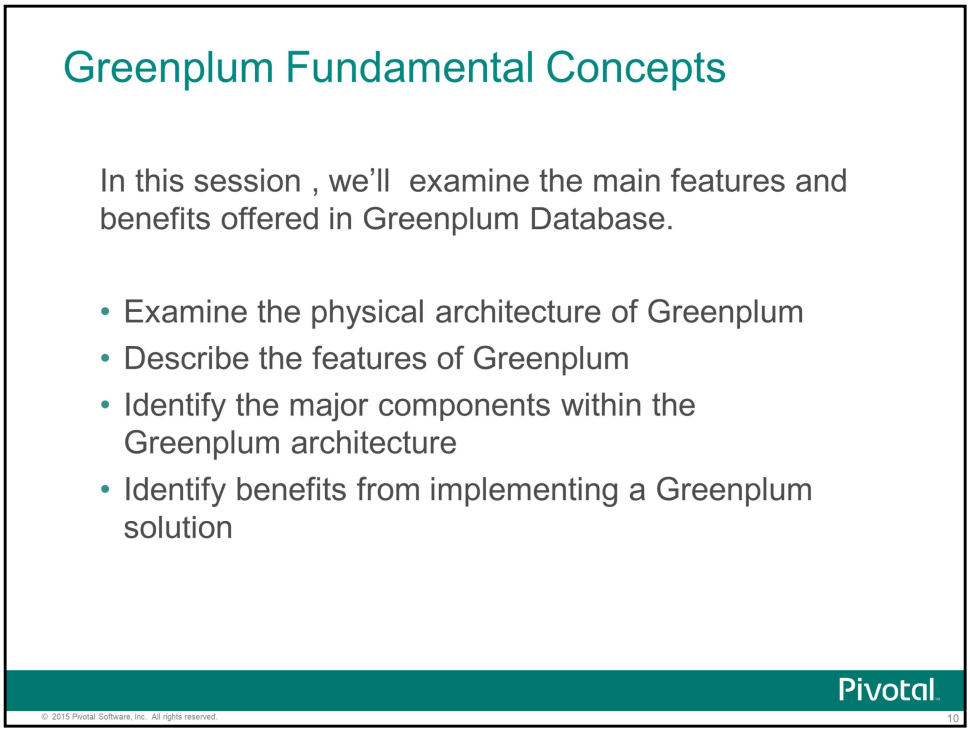
whether we used 10 or 20 or 50 interns. Anytime he wants to ask a question of the business cards, my colleague does not need to know how many interns or who has which data. Once the cards are

distributed, the parallelism is transparent to him.

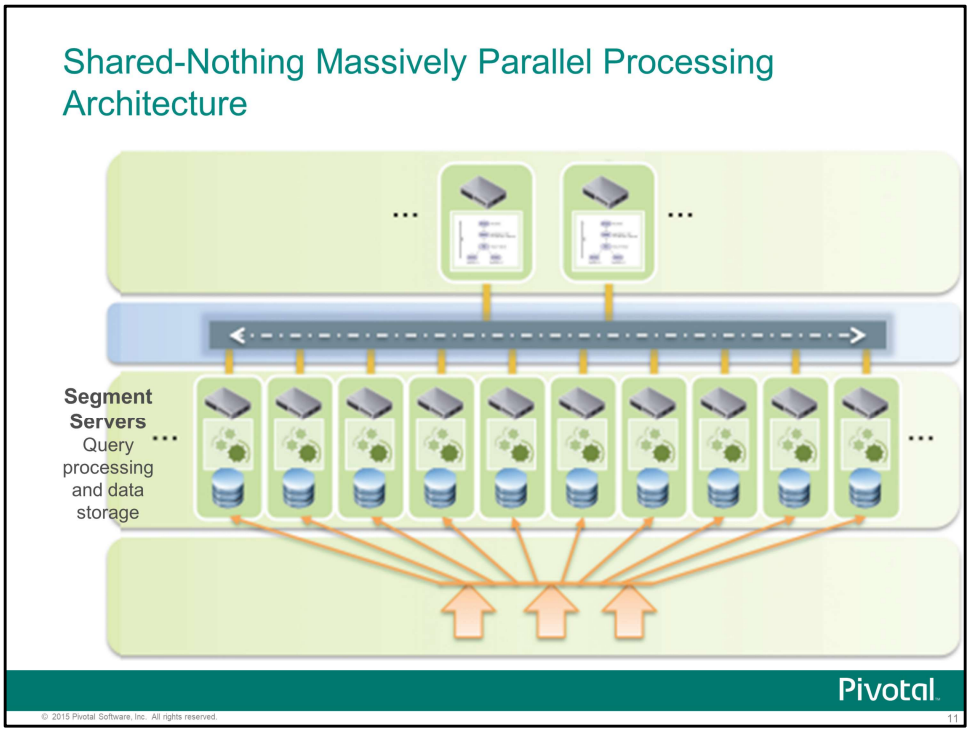
What we have created is an analogue of an MPP (or massively parallel processing)

database.

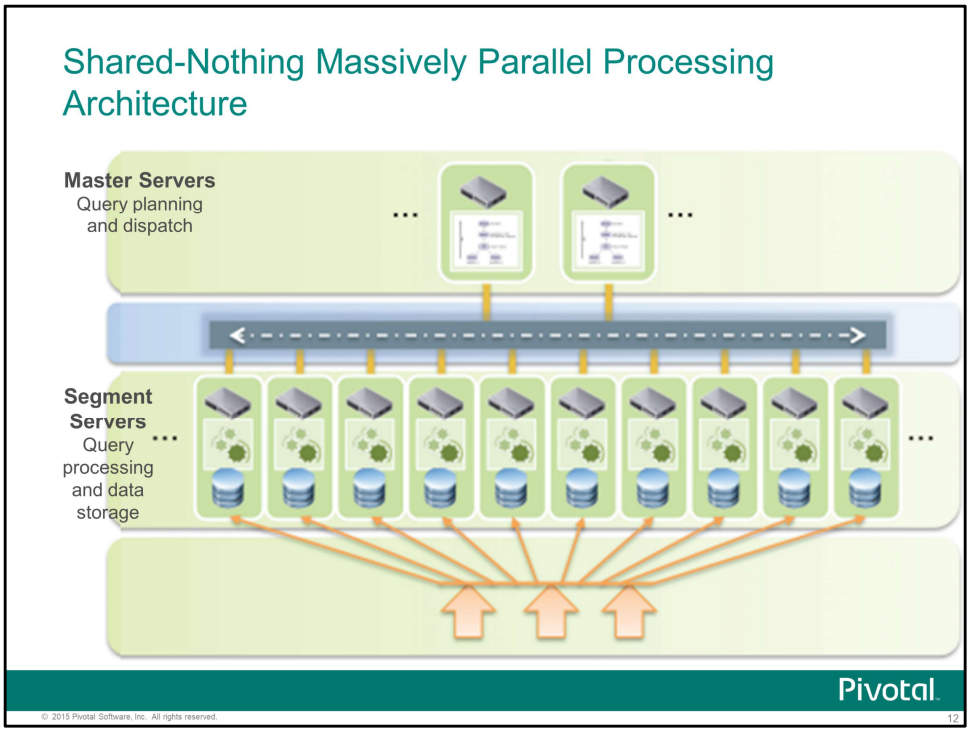
Disclaimer:  No Interns were harmed in the processes mentioned above.



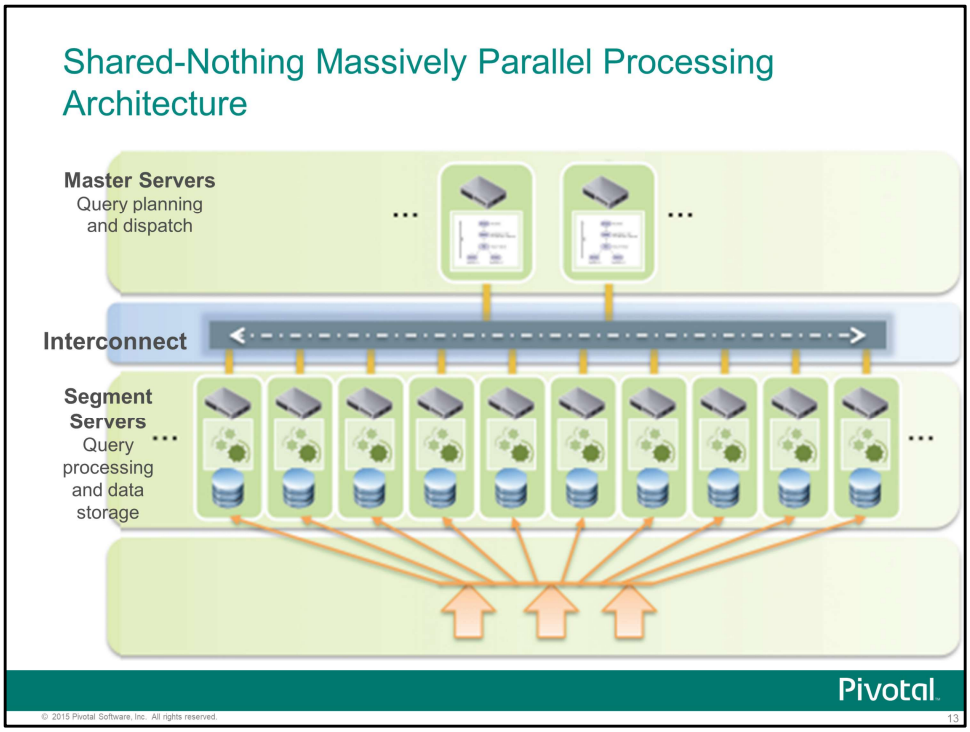

With that in mind, in this lesson, you explore the features and benefits offered in Greenplum Database. You also examine the high-level architecture to understand why Greenplum Database successfully handles mission critical Big Data analytics.



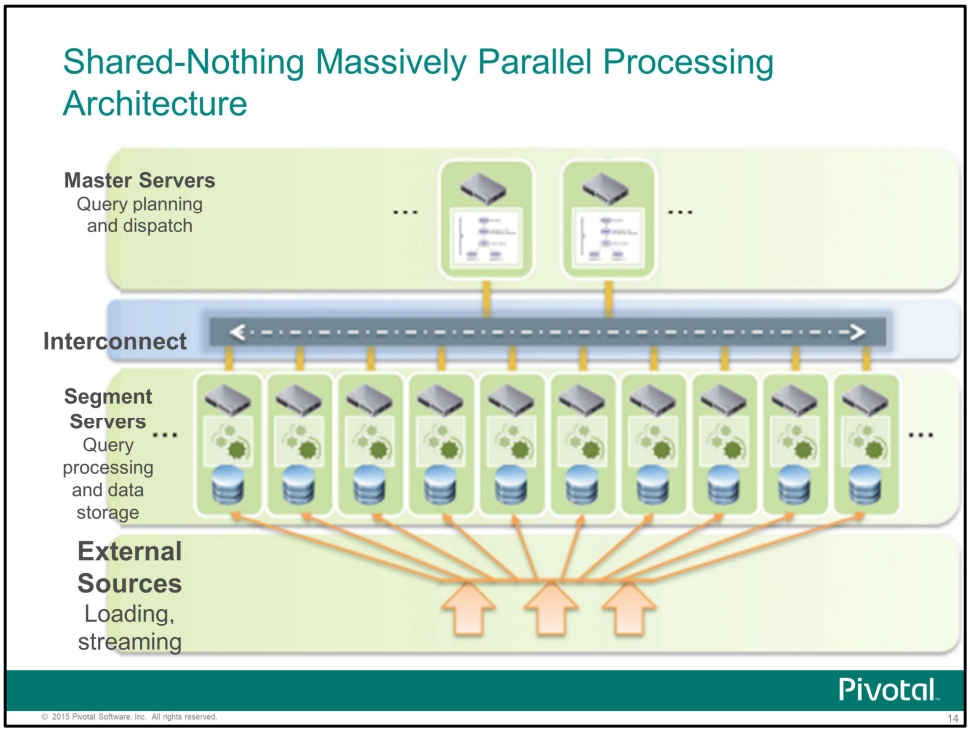

In GPDB, the data is divided into shards or segments  Think of a segment as some portion of the data and the OS PostgreSQL processes needed to analyze the data. .  Many of these segments run on a single host called a segment server.  These are usually small Linux servers with two multi core processors, a sizeable amount of memory, and most important, their own non-shared disks.  The minimum number of segment hosts is usually 4 in a cluster  and we have customers with over one hundred.



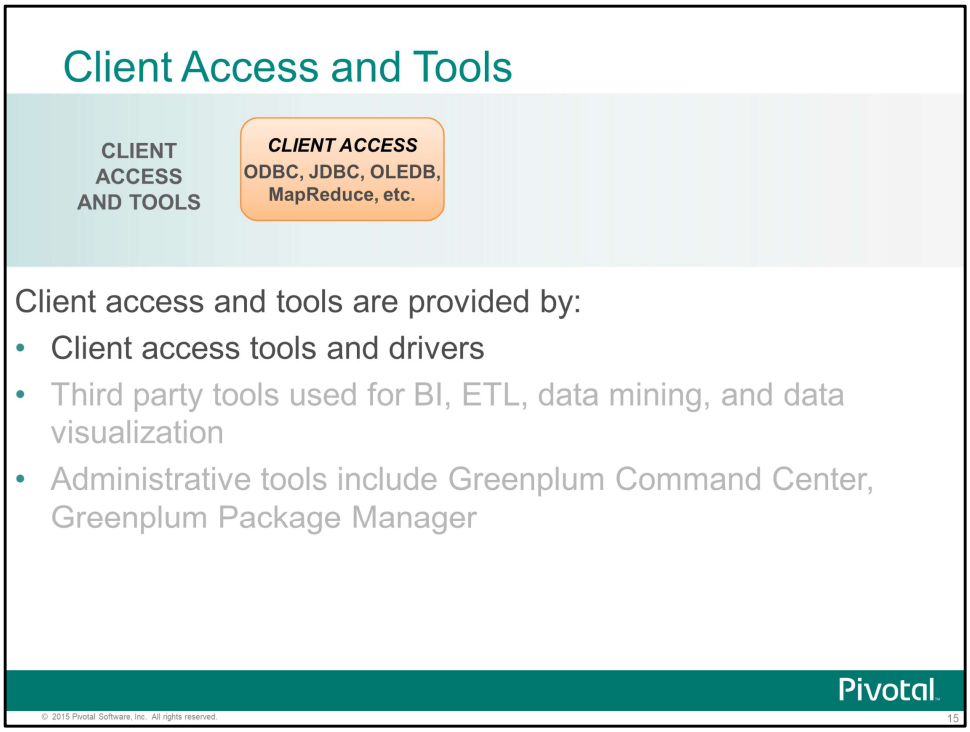

But users never access these segment servers directly. They speak to the Master Server <CLICK>, which has a Postgres instance running with the metadata about the GPDB instance. All of the user data resides on disks in the segment servers, but the metadata lives on disks on the master. The Master parses the query, develops a work plan, and then hands that off to the segment servers and returns a result. There is a Standby Master should the master fail.




In order to facilitate communication, there is a private interconnect or network between the segment servers and master. It is important that this is not part of a public network as adequate bandwidth and latency on this network are necessary for good performance. For example, in doing JOINs, it's often the case that data moves across this interconnect from one segment to another. If the network is being shared with non-GPDB users, performance will likely suffer.

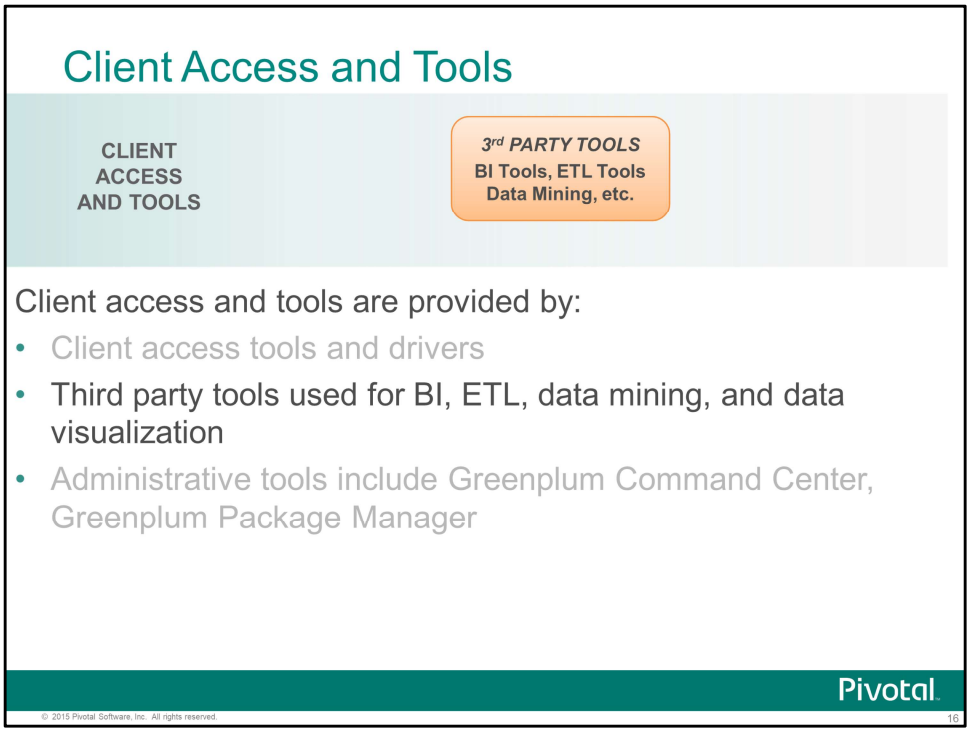



But some other servers are also plumbed into the private network  For speed of data loading, we can put raw data on these servers and then load data in parallel across all the in what is called “Scatter Gather”.  We’ll discuss this in more detail in another section.

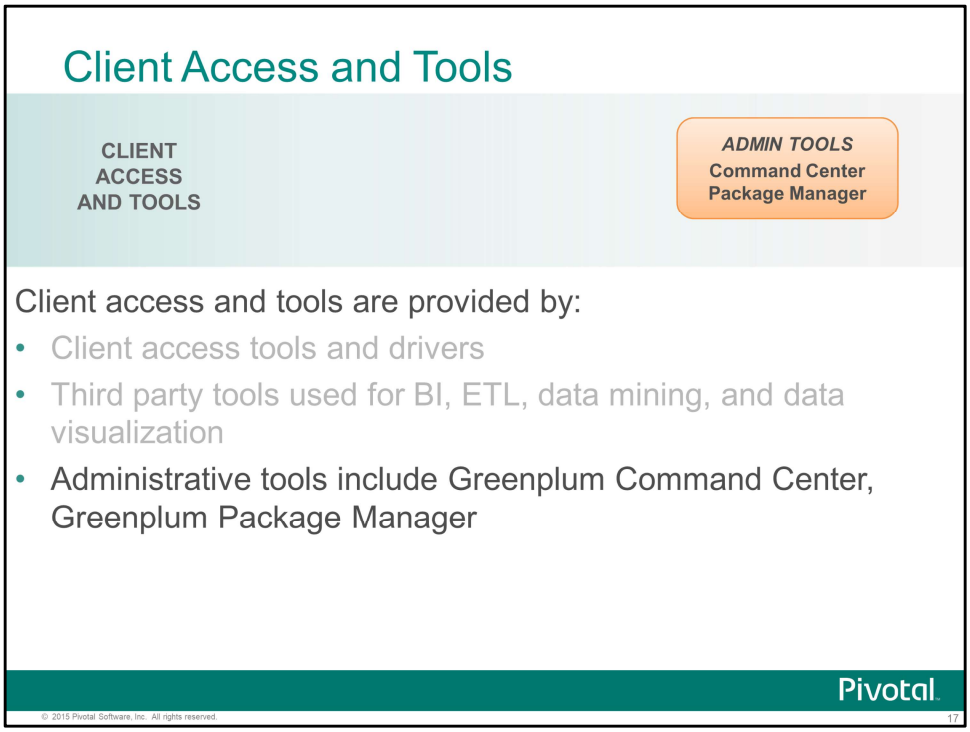



How does this work in practice. Users and administrators have access to Greenplum through a variety of tools:

- **Client access** –You don't need special access tools to use GPDB. Tools and drivers such as ODBC, JDBC, and OLEDB, can be used to access the Greenplum database. You can access your environment with tools such as such as pgAdmin 3. pgAdmin 3 is the most popular and feature-rich Open Source administration and development platform for PostgreSQL. Many use psql, the tools familiar to PostgreSQL users to speak to a GPDB.



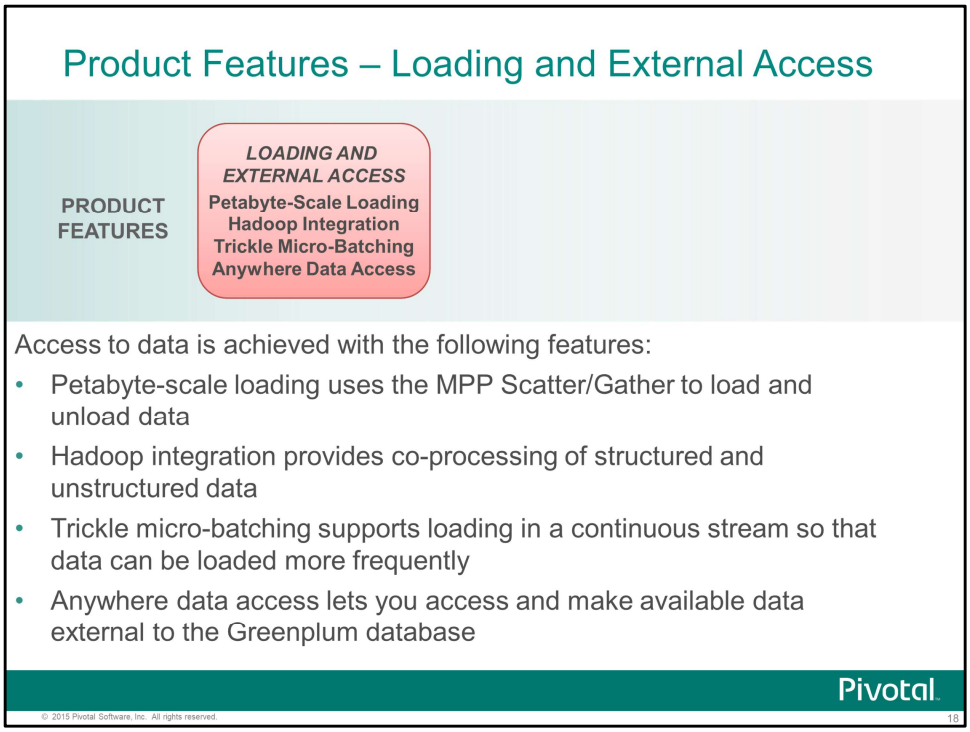

- **Third party tools** –Because GPDB can speak JDBC and ODBC, business intelligence tools, ETL tools, applications for data mining and data visualization can also gain access to the Greenplum database because to them it looks just like a PostgreSQL database. Some vendors have produced connectors that make use of some of GPDB's parallel features, but in a way completely transparently to the users.




- **Administrative tools** – Greenplum Command Center lets administrators manage and monitor the state of the system and workloads, including system metrics and query details on the system. Command Center provides a dashboard for managing and monitoring the system and database, along with queries. You can drill down into a query's detail and plan to understand its performance. Greenplum's Workload Manager, part of GCC, allows rule based control of queries, preventing runaways and throttling down other.

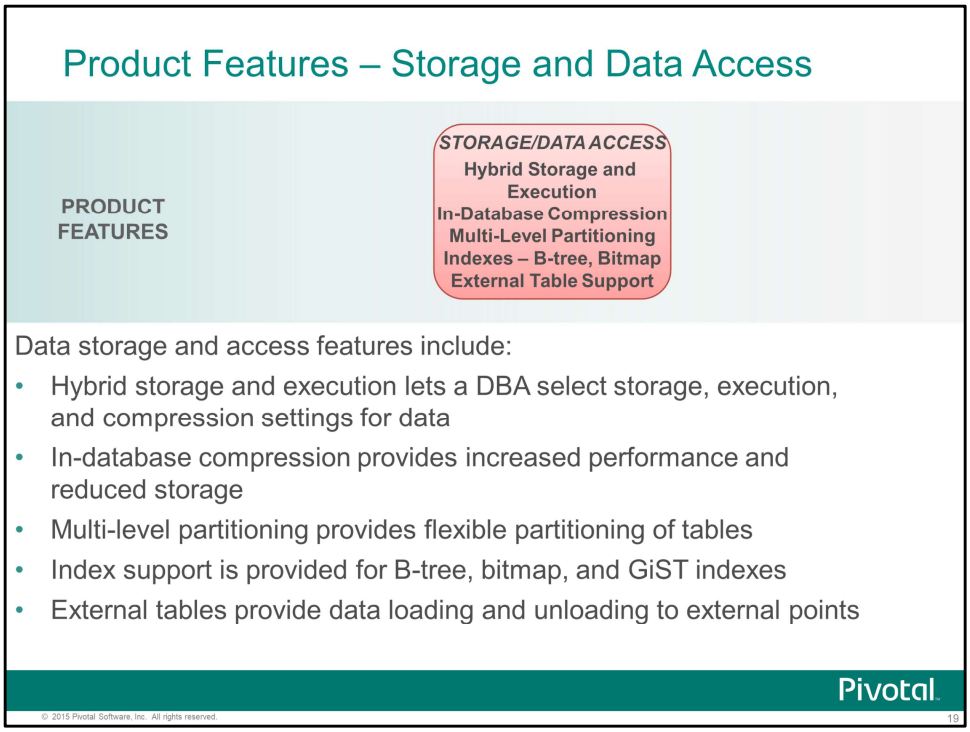
- Greenplum Package Manager lets you install additional supported languages and packages like PostGIS, a standard GIS standard, through a package management utility.

  .



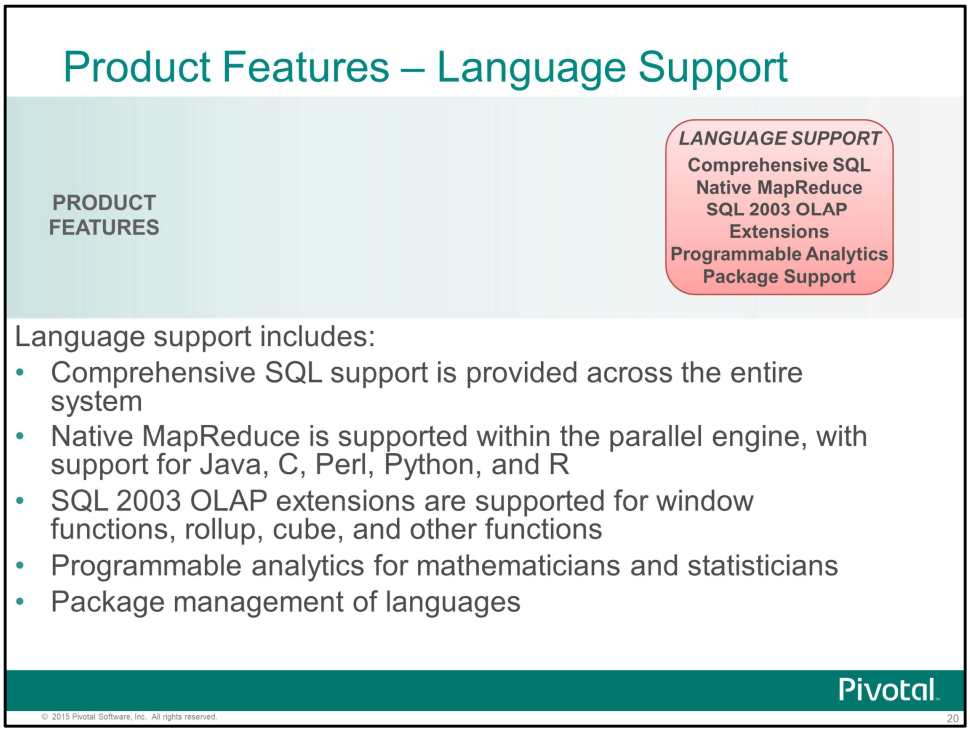

To load data and access external data, Greenplum offers the following features:

- **Petabyte-scale loading** – Using the MPP Scatter/Gather streaming technology, Greenplum can perform high-performance loading and unloading of data. With each additional node in the cluster, the speed at which the loads, parallel data digest, and unloads, parallel data output, are performed increases linearly.
- **Hadoop integration** Greenplum Database provides high performance parallel import and export of data from Hadoop clusters.
- **Trickle micro-batching** – When loading a continuous stream, trickle micro-batching allows data to be loaded at frequent intervals, such as every five minutes, while maintaining extremely high data ingest rates.
- **Anywhere data access** – Data external to the Greenplum database can be accessed, regardless of their location, format, or storage medium. Greenplum allows you to define external tables that access this data and makes it available for reads or writes.



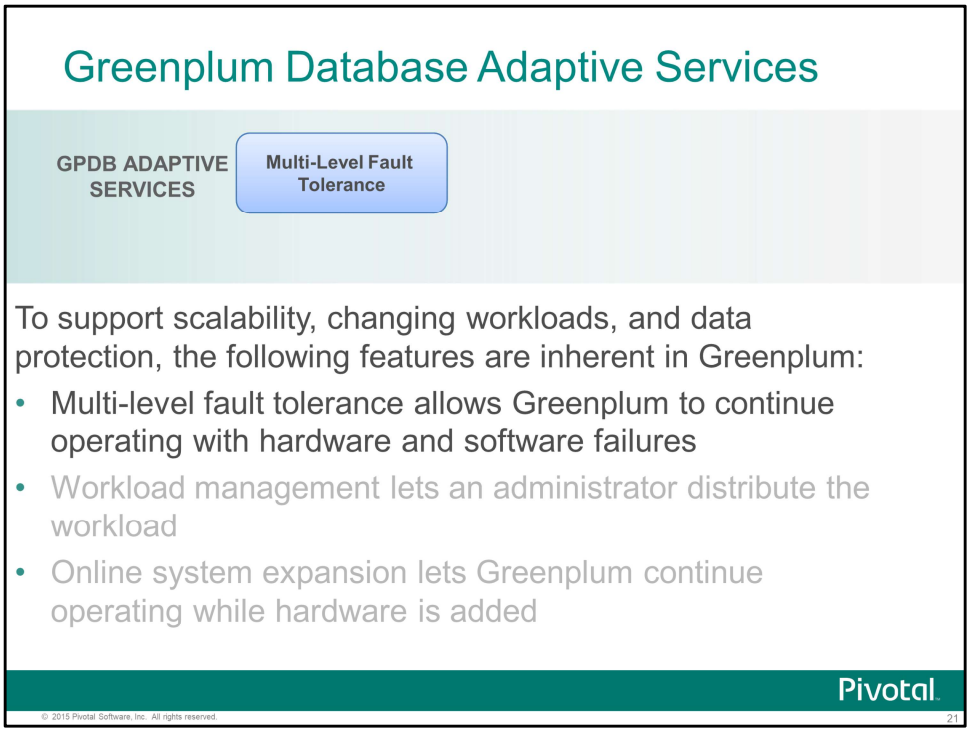

Data storage and access features include:

- **Hybrid storage and execution** – For each table or partition of a table, the database administrator can select the storage, execution, and compression settings that suit the way that table will be accessed. This feature includes the choice of row- or column-oriented storage and processing for any table or partition. This leverages Greenplum's Polymorphic Data Storage technology and allows for tiered storage, where the database administrator can define which data will have lighter compression to allow for faster access and which is not accessed as frequently.
- **In-database compression** – Increased performance and reduced storage can be achieved with in-database compression. By reducing the amount of disk space data takes up, you see an increase in effective I/O performance. In-database compression allows for the storage of years of data, economically. This allows you to get into a true discussion of compliance, e-discovery, and regulatory issues, where you can pull data from previous years quickly. You may not be able to query as quickly, depending on your storage plan, but you will be able to more quickly access data that hasn't been moved off to slow storage or tape.
- **Multi-level partitioning** – With multi-level partitioning, you have flexible partitioning of tables based on date, range, or value. Partitioning is specified using DDL and allows an arbitrary number of levels. The query optimizer will automatically prune unneeded partitions from the query plan.




Powerful language support gives developers flexibility in how to approach Greenplum. Language support is provided for:

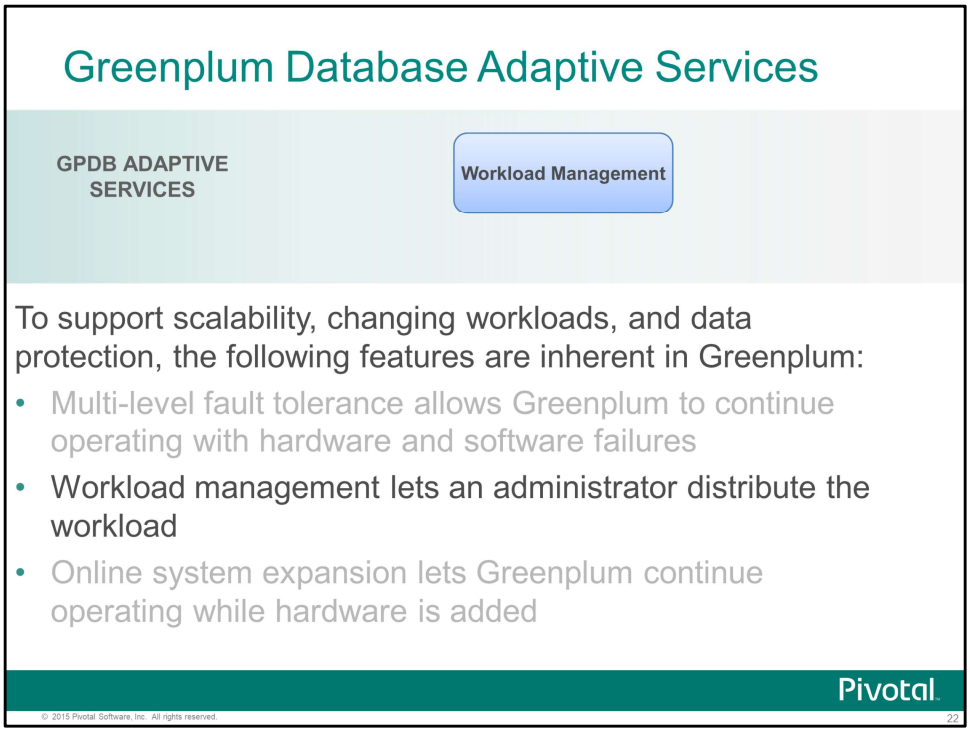
- **Comprehensive SQL** – Greenplum offers comprehensive SQL-92 and SQL-99 support with SQL 2003 OLAP extensions. All queries are parallelized and executed across the entire system.
- **Native MapReduce** – MapReduce has been proven as a technique for high-scale data analysis by Internet leaders such as Google and Yahoo. Greenplum Database natively runs MapReduce programs within its parallel engine. Languages supported include Java, C, Perl, Python, and R.
- **SQL 2003 OLAP Extensions** – Greenplum provides a fully parallelized implementation of SQL recently added OLAP extensions. This includes full standard support for window functions, rollup, cube, and a wide range of other expressive functionality.
- **Programmable Analytics** – With programmable analytics, Greenplum offers a new level of parallel analysis capabilities for mathematicians and statisticians, with support for R, linear algebra, and machine learning primitives. Greenplum also provides extensibility for functions written in Java, C, Perl, or Python.
- **Package Support** – Greenplum also incorporates package support to provide turn key analytic extensions that allows you to more easily manage your language extensions.




Scalability, workload management, and fault tolerance are features that allow Greenplum to adapt to a changing environment, increase uptime, and scale storage and compute power.

- **Multi-level fault tolerance** – Using multiple levels of fault tolerance and redundancy, Greenplum can continue operating in the face of hardware and software failures. Mirrors for the master and segments help to protect against data loss as well as database operation loss. The interconnect provides redundant access to all nodes, the master, standby master, and any other components connected to the switch.
- **Workload management** – The database administrator has administrative control over determining system resources to users and queries. User-based resource queues automatically manage the flow of work to the databases from defined users. Query prioritization allows control of runtime query prioritization to ensure queries have appropriate access to resources. This allows you to prevent one query from hogging all system resources and potentially starving other queries out of these resources. This also allows you to redistribute resources based on user loads.
- **Online system expansion** – Servers can be added to not only increase storage capacity, but also to increase processing power and loading performance. The database can remain online while the expansion process takes place in the background. Due to the implementation of the shared nothing, MPP design, increasing the number of nodes in the cluster increases performance and capacity linearly for Greenplum. Support for dynamic provisioning means you can add onto

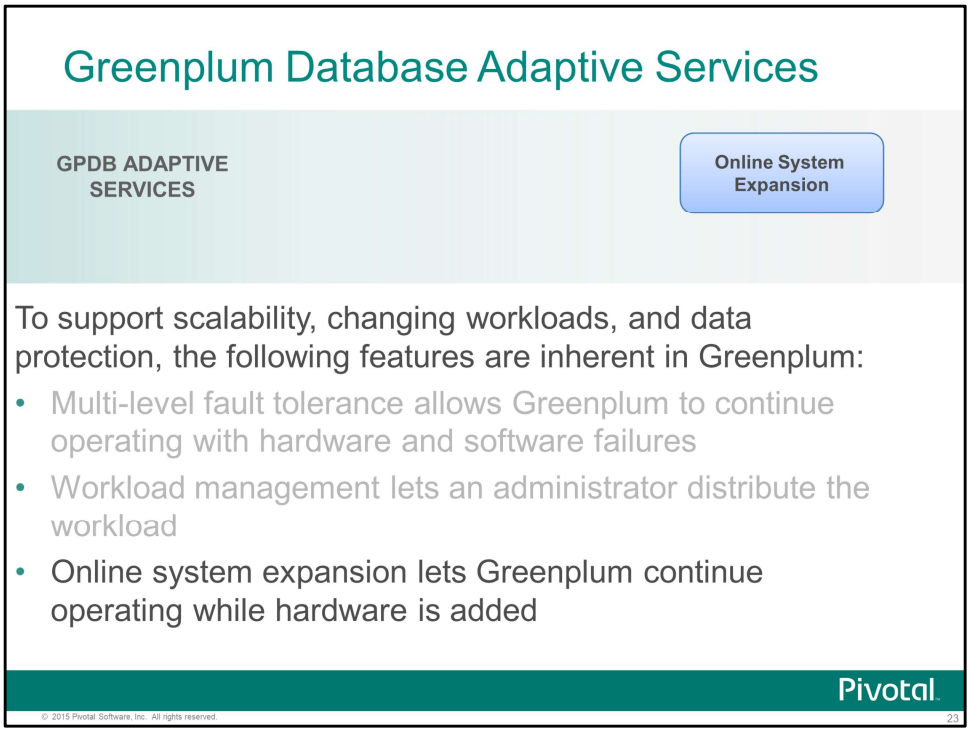
existing configurations without having to replace existing configurations.




Scalability, workload management, and fault tolerance are features that allow Greenplum to adapt to a changing environment, increase uptime, and scale storage and compute power.

- **Multi-level fault tolerance** – Using multiple levels of fault tolerance and redundancy, Greenplum can continue operating in the face of hardware and software failures. Mirrors for the master and segments help to protect against data loss as well as database operation loss. The interconnect provides redundant access to all nodes, the master, standby master, and any other components connected to the switch.
- **Workload management** – The database administrator has administrative control over determining system resources to users and queries. User-based resource queues automatically manage the flow of work to the databases from defined users. Query prioritization allows control of runtime query prioritization to ensure queries have appropriate access to resources. This allows you to prevent one query from hogging all system resources and potentially starving other queries out of these resources. This also allows you to redistribute resources based on user loads.
- **Online system expansion** – Servers can be added to not only increase storage capacity, but also to increase processing power and loading performance. The database can remain online while the expansion process takes place in the background. Due to the implementation of the shared nothing, MPP design, increasing the number of nodes in the cluster increases performance and capacity linearly for Greenplum. Support for dynamic provisioning means you can add onto

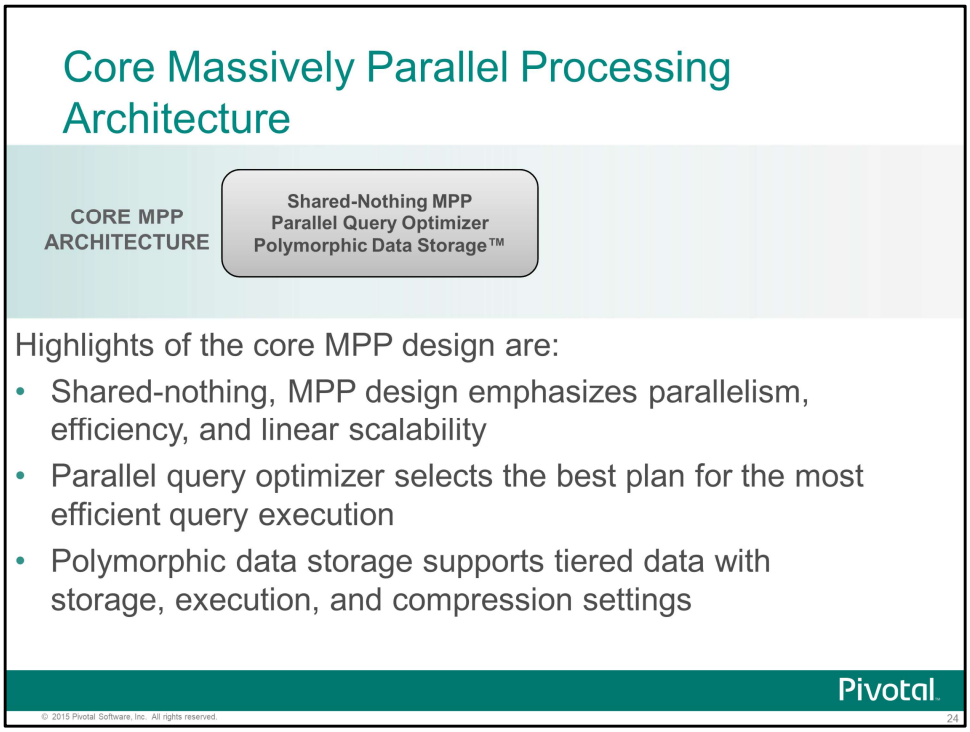
existing configurations without having to replace existing configurations.




Scalability, workload management, and fault tolerance are features that allow Greenplum to adapt to a changing environment, increase uptime, and scale storage and compute power.

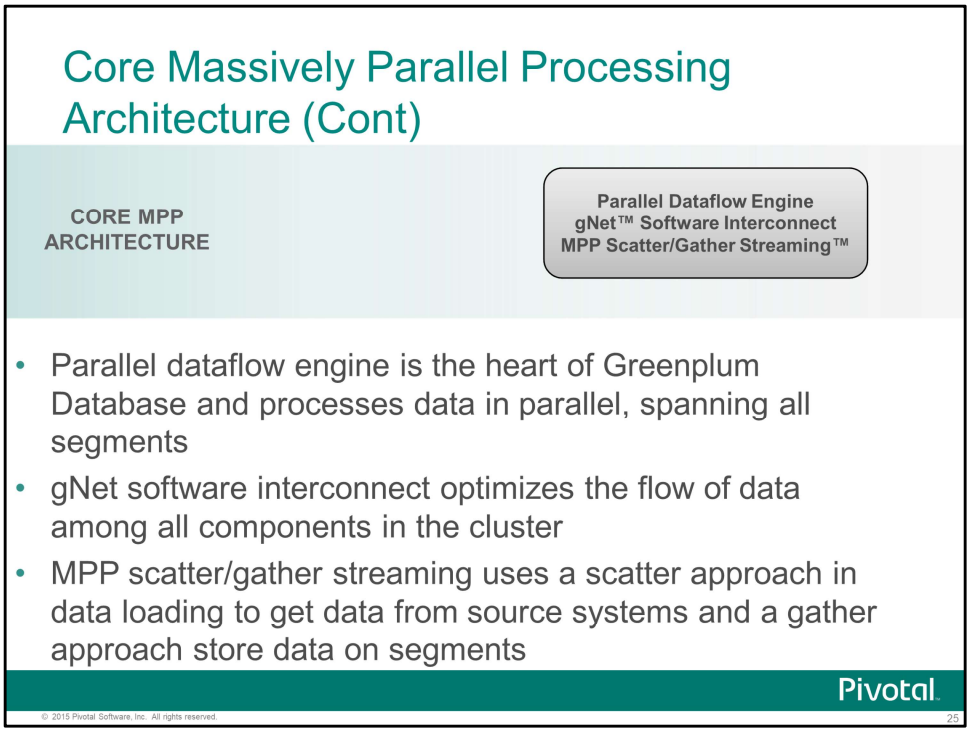
- **Multi-level fault tolerance** – Using multiple levels of fault tolerance and redundancy, Greenplum can continue operating in the face of hardware and software failures. Mirrors for the master and segments help to protect against data loss as well as database operation loss. The interconnect provides redundant access to all nodes, the master, standby master, and any other components connected to the switch.
- **Workload management** – The database administrator has administrative control over determining system resources to users and queries. User-based resource queues automatically manage the flow of work to the databases from defined users. Query prioritization allows control of runtime query prioritization to ensure queries have appropriate access to resources. This allows you to prevent one query from hogging all system resources and potentially starving other queries out of these resources. This also allows you to redistribute resources based on user loads.
- **Online system expansion** – Servers can be added to not only increase storage capacity, but also to increase processing power and loading performance. The database can remain online while the expansion process takes place in the background. Due to the implementation of the shared nothing, MPP design, increasing the number of nodes in the cluster increases performance and capacity linearly for Greenplum. Support for dynamic provisioning means you can add onto

existing configurations without having to replace existing configurations.

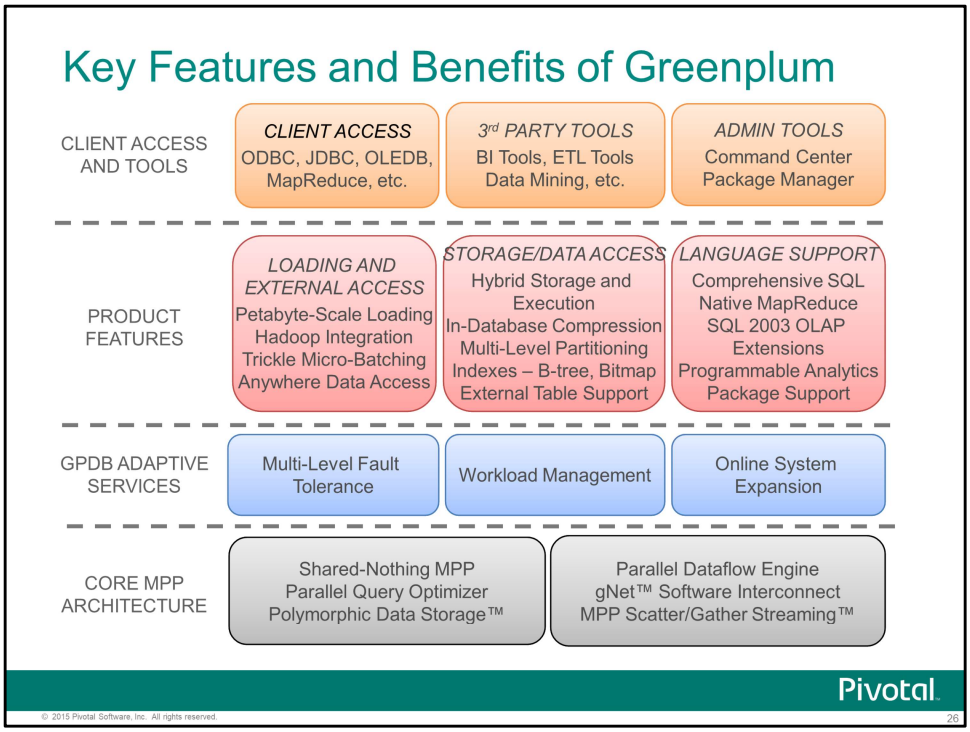



The core massively parallel processing architecture highlights several major features:

- **Shared-nothing, MPP design** – The shared-nothing, massively parallel processing architecture utilized by Greenplum incorporates the highest level of parallelism and efficiency to handle complex business intelligence and analytical processing. Greenplum takes advantage of the available hardware in its environment to ensure that data is automatically distributed and query workload is parallelized. Each unit, or segment, within the environment, acts as a self-contained database management system that owns and manages a distinct portion of the overall data. While the data and execution are parallelized, all nodes within a Greenplum environment work together in a highly coordinated fashion.
- **Parallel query optimizer** – When it receives a query, the master server uses a cost-based optimization algorithm to evaluate a vast number of potential plans and selects the one it believes is the most efficient. It does this by taking a global view of execution across the cluster and factors in the cost of moving data between nodes. By taking a global view, you obtain more predictable results than an approach which requires replanning at each node.
- **Polymorphic data storage** – A polymorphic data storage allows customers to choose the storage, execution, and compression settings to support row or column oriented storage and retrieval. This lends support to tiered or temperature aware data, where you may opt to store older data as column oriented with deep archival compression on one set of disks, while more recent data is stored with fast and light compression.



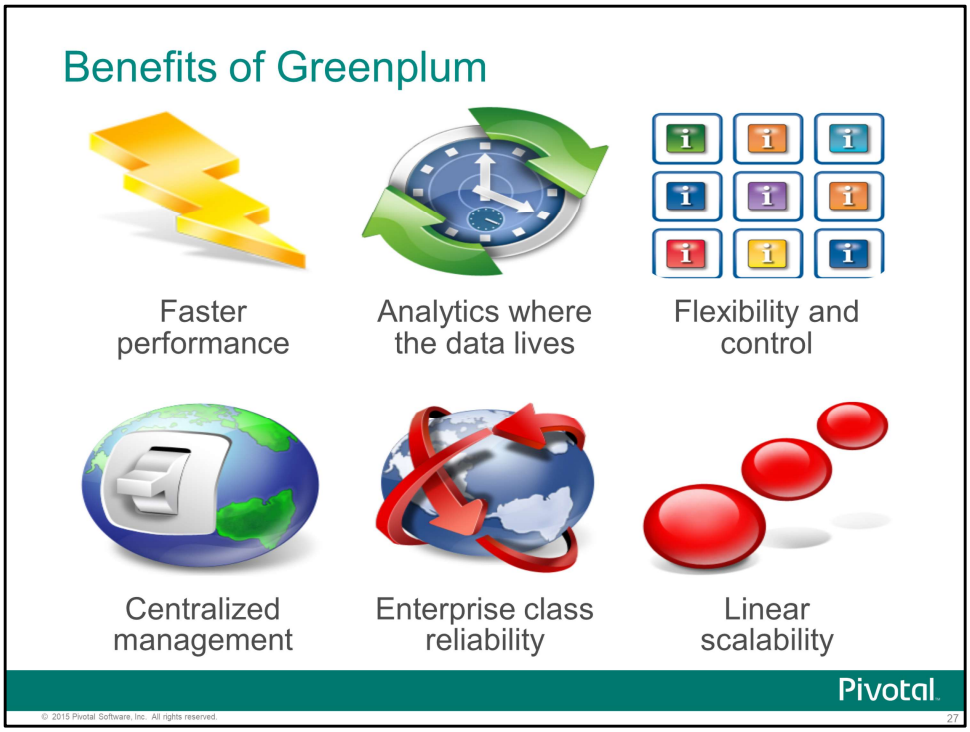

- **Parallel dataflow engine** – The work of processing and analyzing data is performed in the parallel dataflow engine, the heart of Greenplum Database. This optimized parallel processing infrastructure processes data as it flows from disk, external sources, or other segments over the interconnect. The engine is inherently parallel and spans all segments of a cluster. It can effectively scale to thousands of commodity processing cores.
  It is highly optimized at executing both SQL and MapReduce in a massively parallel manner. It has the ability to directly execute all necessary SQL building blocks, including performance-critical operations such as hash-join, multistage hash-aggregation, SQL 2003 windowing, and arbitrary MapReduce programs.
- **gNet software interconnect** – One of the most critical components in Greenplum, the gNet interconnect optimizes the flow of data to allow for continuous pipelining of processing without blocking on all nodes of a system. It leverages commodity Gigabit Ethernet and 10GigE switch technology to efficiently pump streams of data between motion nodes during query plan execution. It utilizes pipelining, the ability to begin a task before its predecessor task has completed, to ensure the highest-possible performance.
- **MPP scatter/gather streaming** – Using the MPP scatter/gather streaming, Greenplum is able to achieve data loads of more than 4 terabytes per hour with negligible impact on concurrent database operations. Using a parallel-everywhere approach to data loading, data is scattered from all source systems across hundreds or thousands of parallel streams to all nodes I the cluster. Each node in the cluster simultaneously gathers the data it is responsible for.




Greenplum is a shared-nothing, massively parallel processing (MPP) architecture designed for business intelligence and analytical processing. It is one of the first open-source databases, based on PostgreSQL, that was made available to enterprise environments. Built to support \Big Data, Greenplum manages, stores, and analyzes terabytes to petabytes of data, with vastly improved performance over traditional relational database management system products.

The logical architecture represents the major features and benefits Greenplum offers. Starting from the bottom of the illustration, you have:

- Core massively parallel processing architecture
- Greenplum adaptive services
- Product features
- Client access and tools




In summary . **[LAST SLIDE FOR THE INTRO/OVERVIEW MODULE]**

Customers who implement Greenplum gain benefits in:

- **Faster performance** – Faster loads and faster queries.
- **Real-time analytics** – Greenplum enables sophisticated queries and ad-hoc analysis with multiple terabytes to petabytes of data. With OLAP queries, you can perform advanced queries without having to use third party tools. In Greenplum, do the analytic where the data lives. Don't ship it to a separate analytics engine.
- **Flexibility and control** – You can decide whether to choose an appliance or your own hardware. This gives you control over the choice of hardware and operating systems, as well as the ability to add capacity and therefore performance, inexpensively.
- **Centralized management** – Centralized management allows ease of configuration through a single central location – the master. In the end, centralized cluster management and administration lowers total cost of ownership (TCO).
- **Enterprise class reliability** – High availability, mirroring on segments and standby and hardware-level mirroring. With multiple levels of redundancy and fail-over, this minimizes downtime.
- **Linear scalability** – Nodes can be expanded on an as needed basis. This allows for predictable, linear performance gains and capacity growth. It is recommended that Professional Services is involved when expanding nodes. Expanding nodes involves

reconfiguring the database to make immediate use of the hardware with preexisting data and newly stored data.

- **Open Source** – the Greenplum database in now in open source. This brings a large developer community and set of eyes on code.