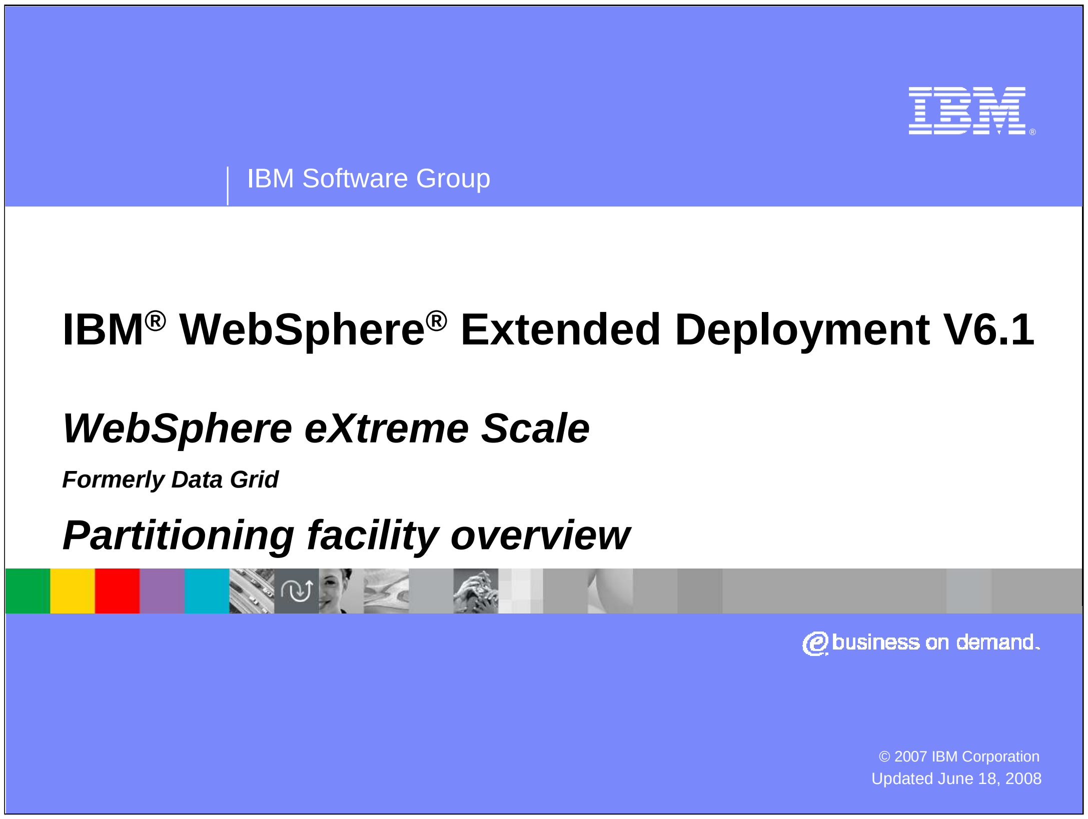


IBM Software Group

IBM® WebSphere® Extended Deployment V6.1

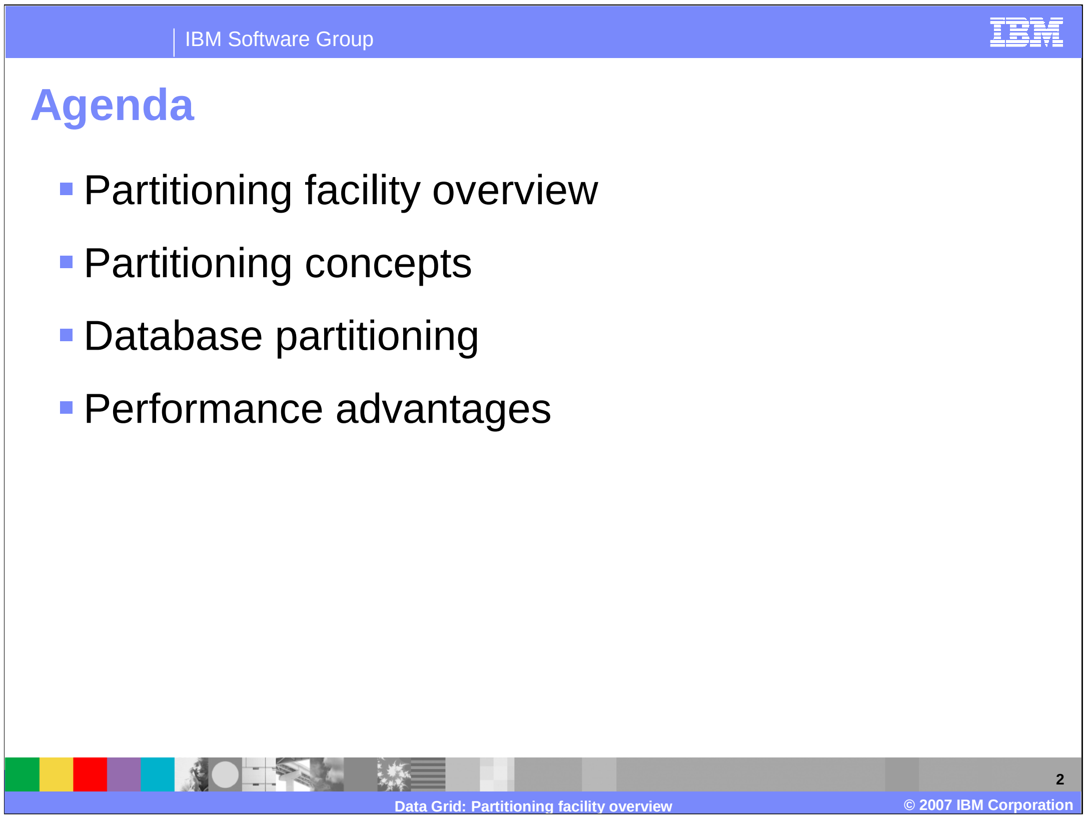
WebSphere eXtreme Scale

Formerly Data Grid

Partitioning facility overview

This presentation will cover the partitioning facility in WebSphere Extended Deployment V6.1

This module was originally recorded for WebSphere Extended Deployment Data Grid, which is now called WebSphere eXtreme Scale. Though the module uses the previous names, the technical material covered is still accurate.



Agenda

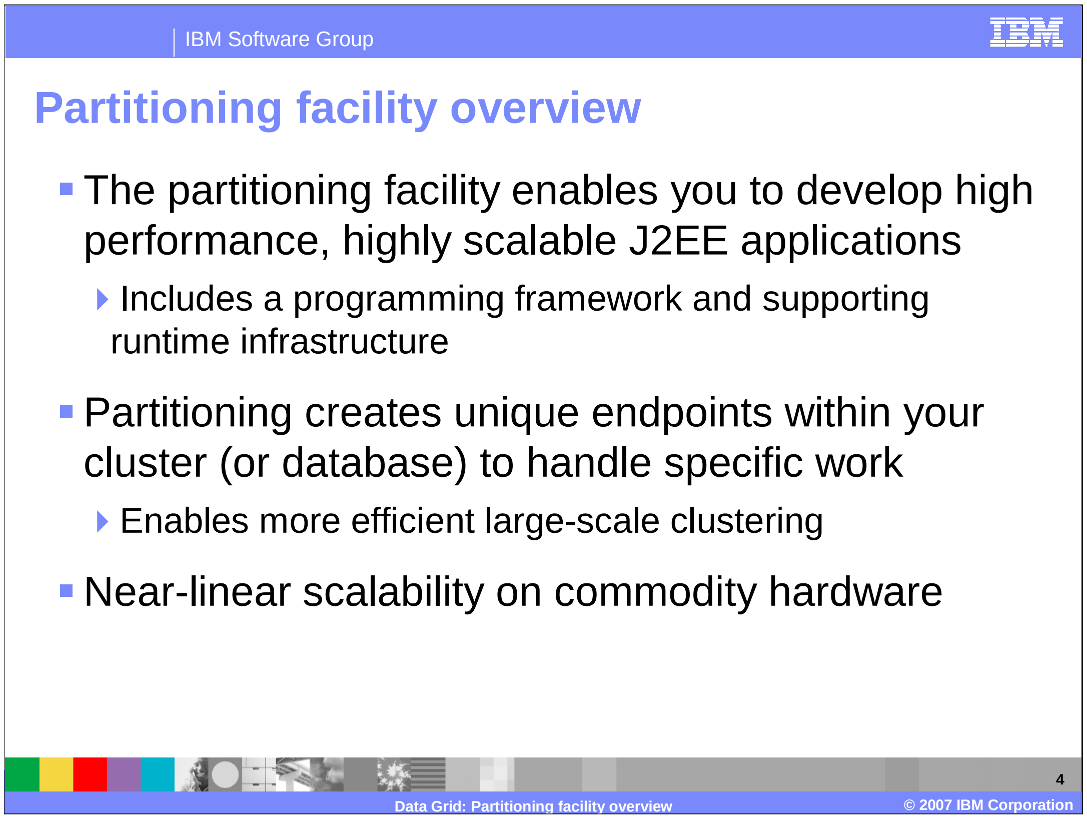
- Partitioning facility overview
- Partitioning concepts
- Database partitioning
- Performance advantages

This presentation will first introduce the partitioning facility through an example scenario that compares a classic Java™ 2 Enterprise Edition (J2EE) application to a partitioned application. It will then cover concepts associated with partitioning to familiarize you with the partitioning facility. Database partitioning and HTTP partitioning will also be discussed. The presentation will end by summarizing the performance advantages of partitioning, and direct you to other resources for learning about the partitioning facility.

Section

Overview

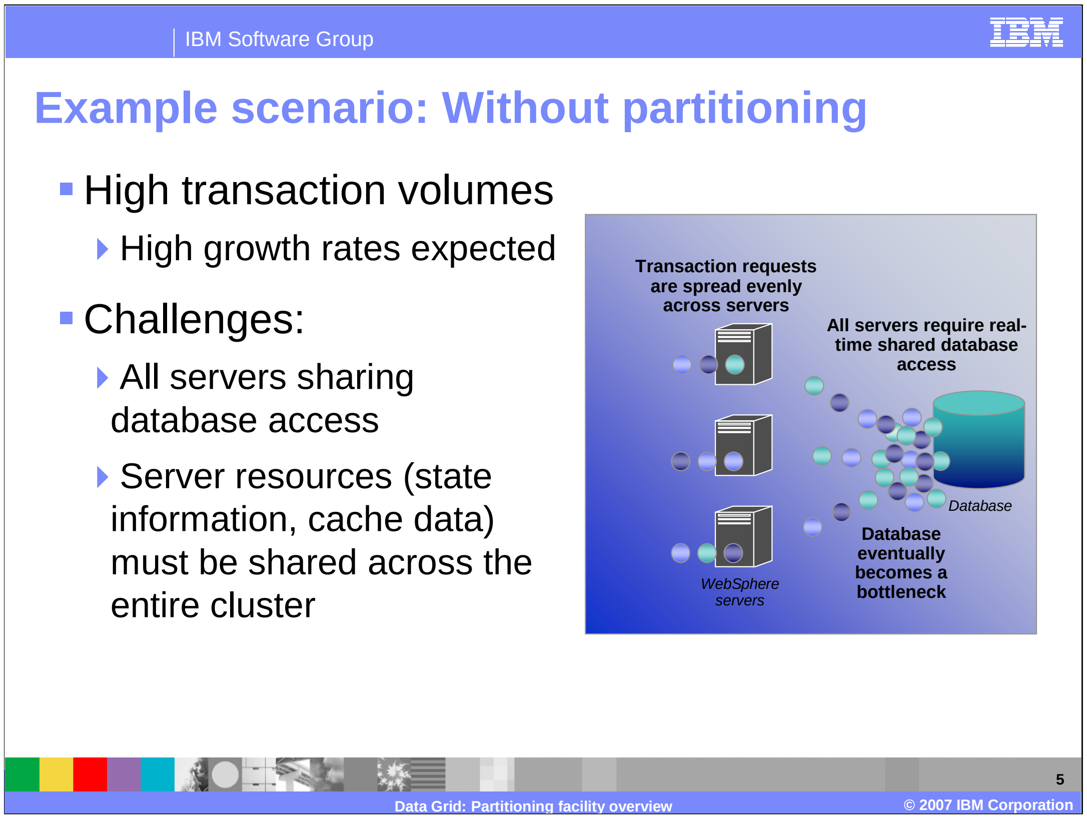
This section will introduce the partitioning facility.



Partitioning facility overview

- The partitioning facility enables you to develop high performance, highly scalable J2EE applications
  - Includes a programming framework and supporting runtime infrastructure
- Partitioning creates unique endpoints within your cluster (or database) to handle specific work
  - Enables more efficient large-scale clustering
- Near-linear scalability on commodity hardware

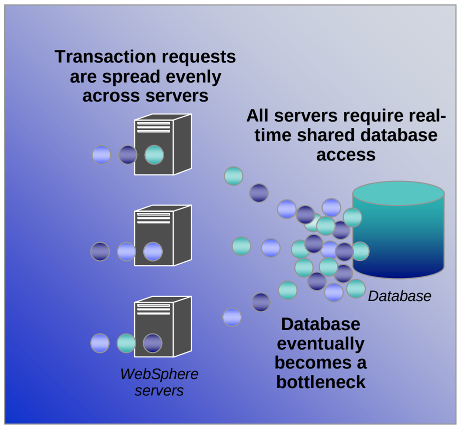
The partitioning facility is a programming framework and runtime infrastructure that enables you to create J2EE applications that scale efficiently to support very high transaction volumes. This is accomplished by creating unique endpoints, or ‘partitions’, within a cluster to handle requests for particular data sets, which can also communicate with a partitioned database. Requests can then be consistently routed to the single known endpoint within the cluster that will handle the specific data set being accessed. The strategy effectively reduces contention for database resources, and avoids some of the traditional challenges of large-scale clustering, for example large-scale replication of cache data. Applications that utilize the partitioning facility can achieve near-linear scalability on commodity hardware.



IBM Software Group

Example scenario: Without partitioning

- High transaction volumes
  - High growth rates expected
- Challenges:
  - All servers sharing database access
  - Server resources (state information, cache data) must be shared across the entire cluster

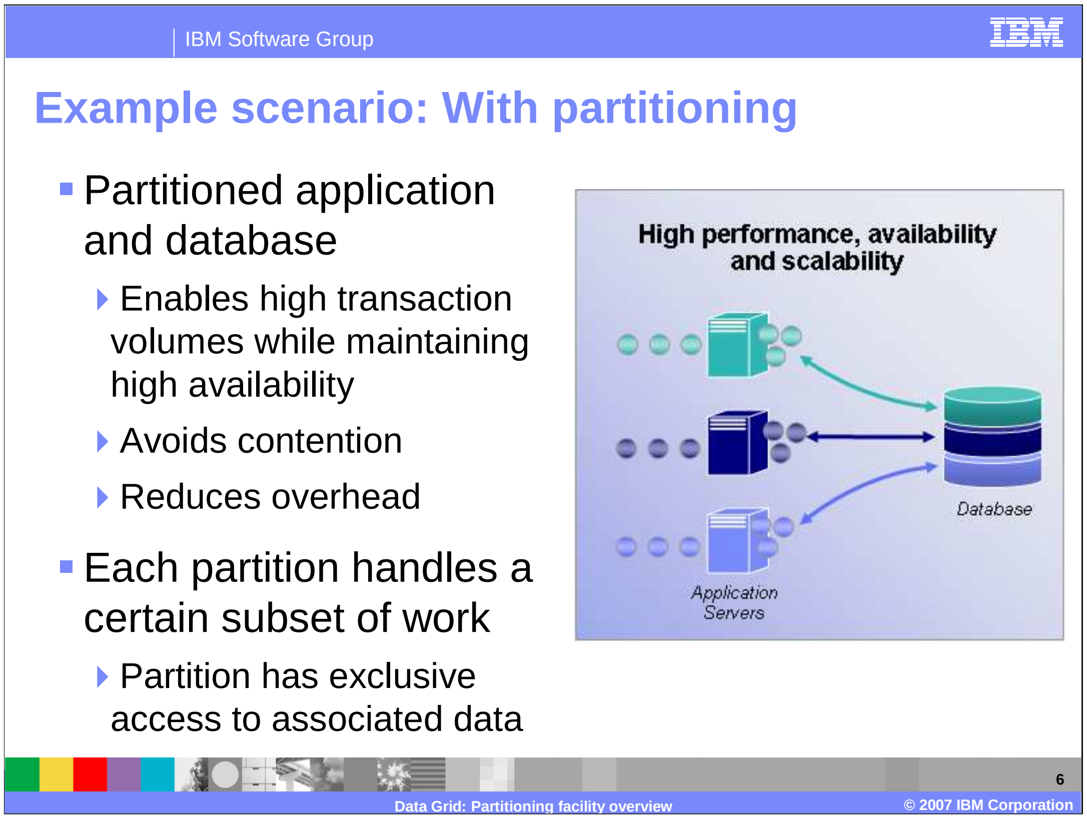

Transaction requests are spread evenly across servers
All servers require real-time shared database access
Database
Database eventually becomes a bottleneck
WebSphere servers

5

Data Grid: Partitioning facility overview

© 2007 IBM Corporation

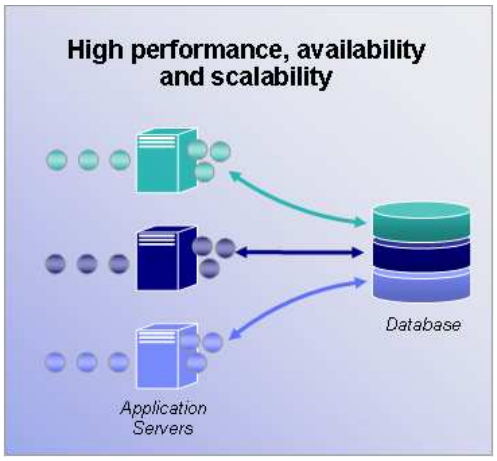
As an example, imagine a traditional environment that is hosting a very high volume application for which traffic is expected to increase over time. Requests are distributed equitably across all of the servers in the cluster. Each of these requests may require a database transaction, which is expensive, and any request that requires a database update will require a lock that prevents other servers from reading the affected data. In this scenario, the database is very likely going to be the performance bottleneck as volume increases. Also, if the servers are caching any data or require shared state information, the overhead of replicating that data can become large as the size of the cluster increases. If the quality of service declines in this environment under increased load, adding more application servers will probably not improve the situation.



IBM Software Group

Example scenario: With partitioning

- Partitioned application and database
  - Enables high transaction volumes while maintaining high availability
  - Avoids contention
  - Reduces overhead
- Each partition handles a certain subset of work
  - Partition has exclusive access to associated data



6

Data Grid: Partitioning facility overview © 2007 IBM Corporation

Partitioning solves this scalability problem. The cluster is divided into several uniquely addressable partitions, based on the data sets they will work with, and the database is also partitioned on the same lines. When a request comes in for a particular set of data, such as a given stock symbol, it is predictably routed to a particular partition. The application server hosting that partition has exclusive access to the data related to the requested stock symbol, and exclusive access to the database instance that holds that data. This means that database read and write activities for the stock symbol ‘ABC’ will not cause contention with a request from another server. Additionally, the application can be designed to cache data in memory more aggressively, resulting in fewer database transactions, since no other servers will be accessing that data. As load on the system increases over time, servers can be added to the environment, spreading the application and database partitions across more servers. This gives more computing resources to each partition, as they are spread more thinly. In this scenario, adding hardware will allow the application to scale efficiently and keep up with the increased demand.

IBM Software Group

Section

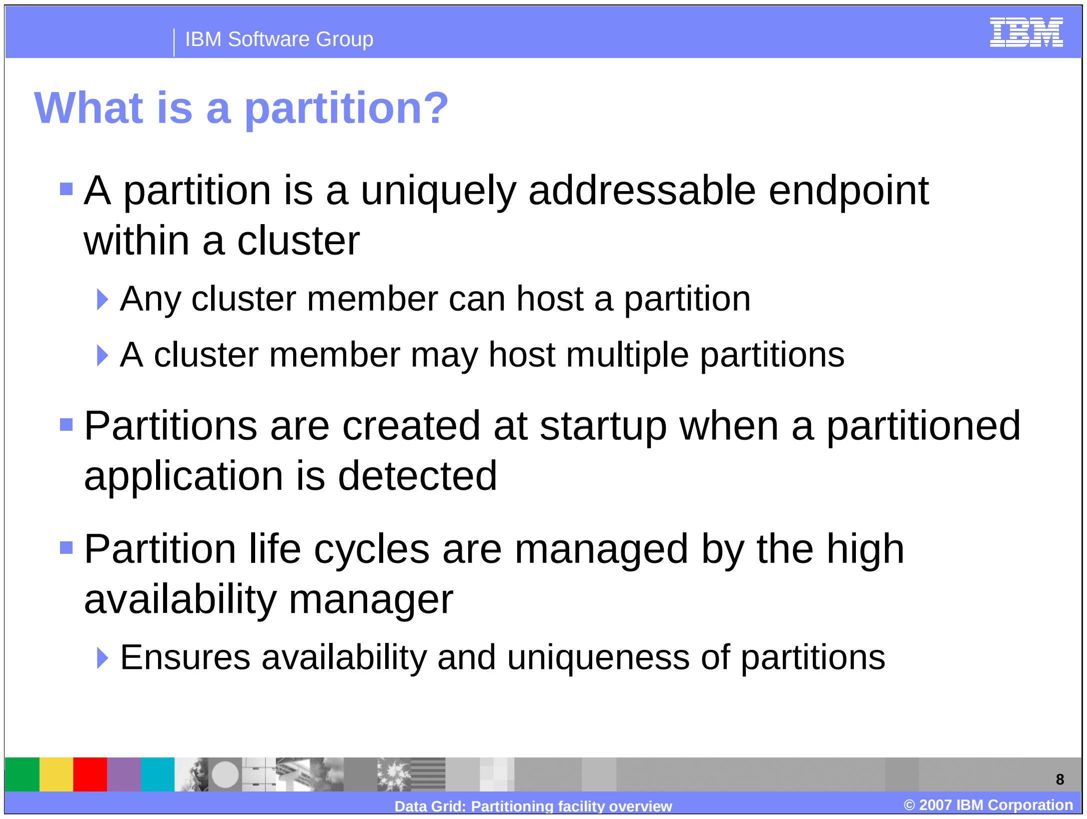
Partitioning concepts

7

Data Grid: Partitioning facility overview

© 2007 IBM Corporation

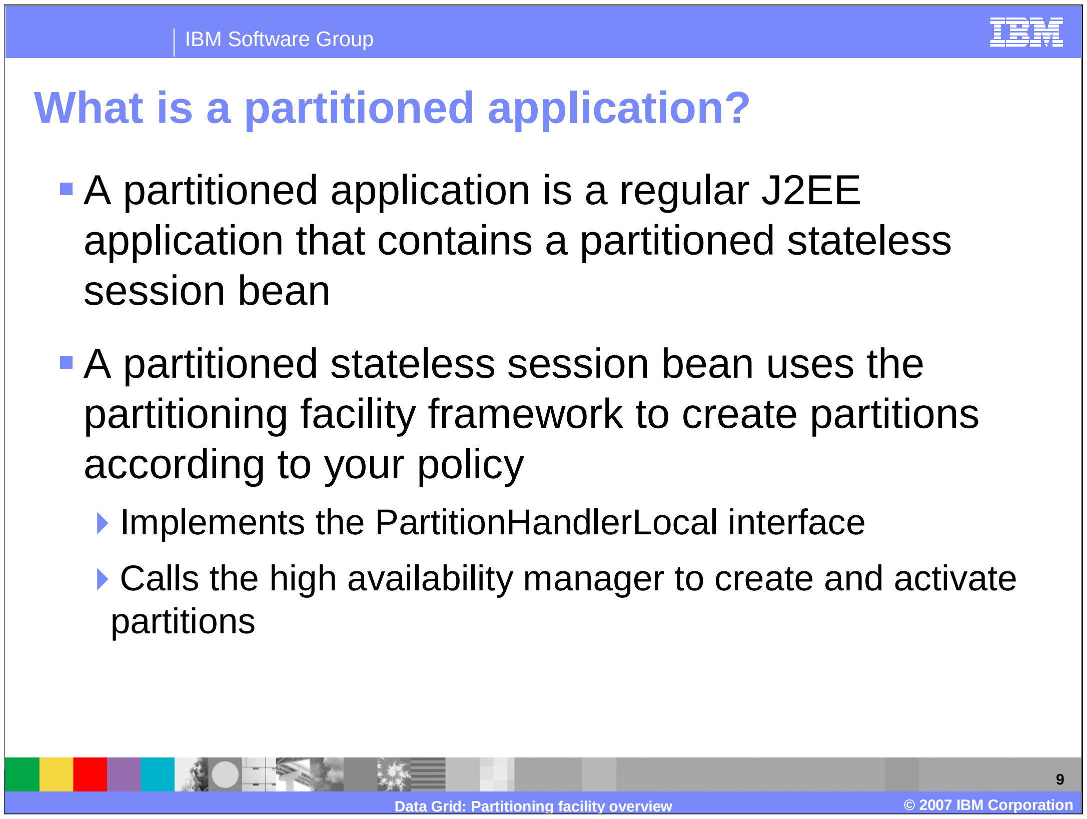
This section will discuss concepts associated with partitioning.



What is a partition?

- A partition is a uniquely addressable endpoint within a cluster
  - Any cluster member can host a partition
  - A cluster member may host multiple partitions
- Partitions are created at startup when a partitioned application is detected
- Partition life cycles are managed by the high availability manager
  - Ensures availability and uniqueness of partitions

A partition is a uniquely addressable endpoint within a cluster. Each cluster member might host multiple partitions, but each partition exists on only one cluster member. Each partition is started as a highly available singleton, managed by the high availability manager infrastructure. This ensures that the partitions remain constantly available while running in only one cluster member at any given time.


IBM Software Group

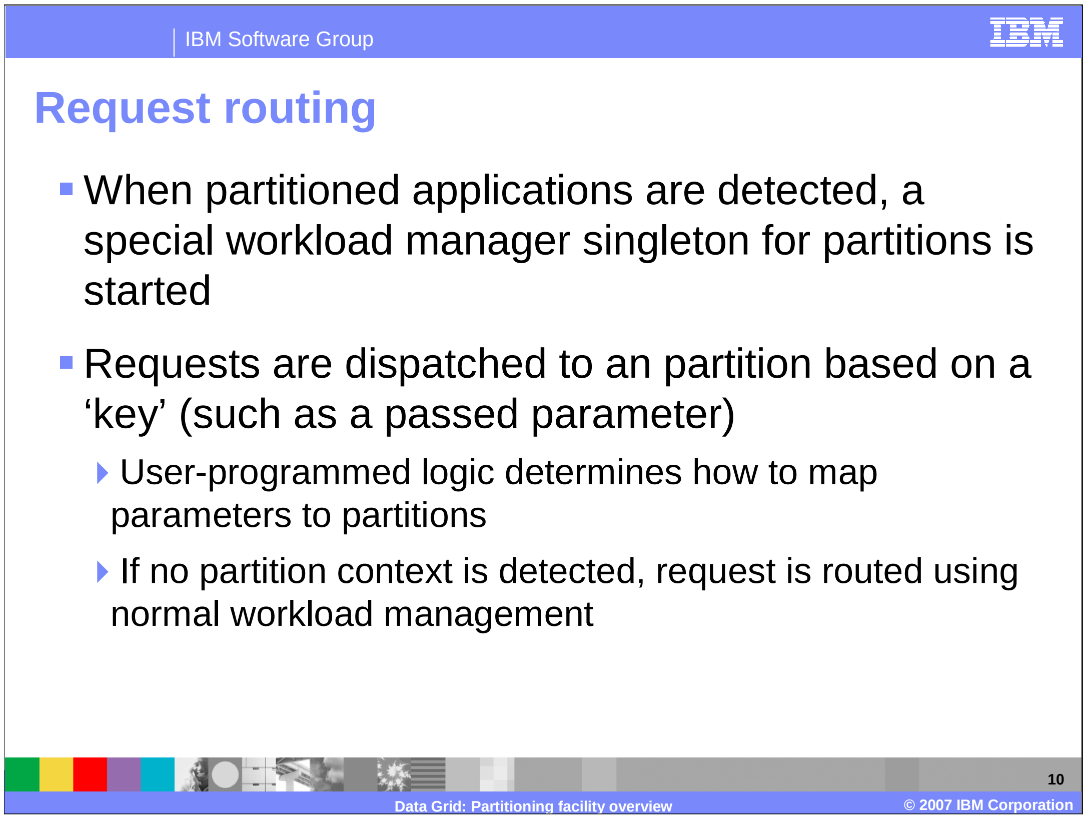
What is a partitioned application?

- A partitioned application is a regular J2EE application that contains a partitioned stateless session bean
- A partitioned stateless session bean uses the partitioning facility framework to create partitions according to your policy
  - Implements the PartitionHandlerLocal interface
  - Calls the high availability manager to create and activate partitions

9

Data Grid: Partitioning facility overview © 2007 IBM Corporation

A partitioned application contains a single partitioned stateless session bean. This bean implement the PartitionHandlerLocal interface, and instruct the partitioning facility as to how partitions should be created and how requests should be mapped to partitions. The WebSphere Extended Deployment runtime identifies partitioned applications by looking for this bean.


IBM Software Group

Request routing

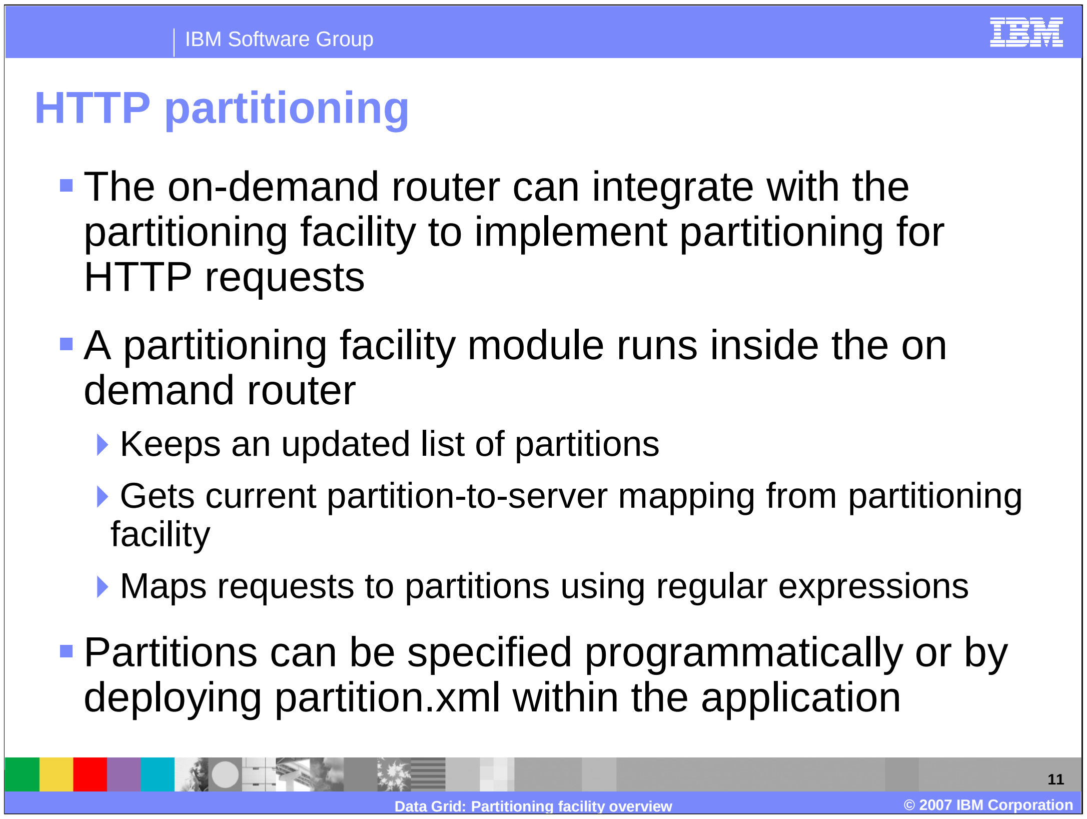
- When partitioned applications are detected, a special workload manager singleton for partitions is started
- Requests are dispatched to an partition based on a ‘key’ (such as a passed parameter)
  - User-programmed logic determines how to map parameters to partitions
  - If no partition context is detected, request is routed using normal workload management

10

Data Grid: Partitioning facility overview

© 2007 IBM Corporation

The WebSphere Extended Deployment runtime provides a separate workload management router for partitioned applications, which is started automatically when a partitioned application is detected. This router locates and routes the request to the appropriate partition based on a partition key. The application defines partition keys and how they are mapped to partitions. The partition key could be a parameter passed to the method being called, or a hash based on such a parameter. If a partition key is not associated with the request, it will be handled by the normal workload manager, just like any other IIOP or HTTP request.


IBM Software Group

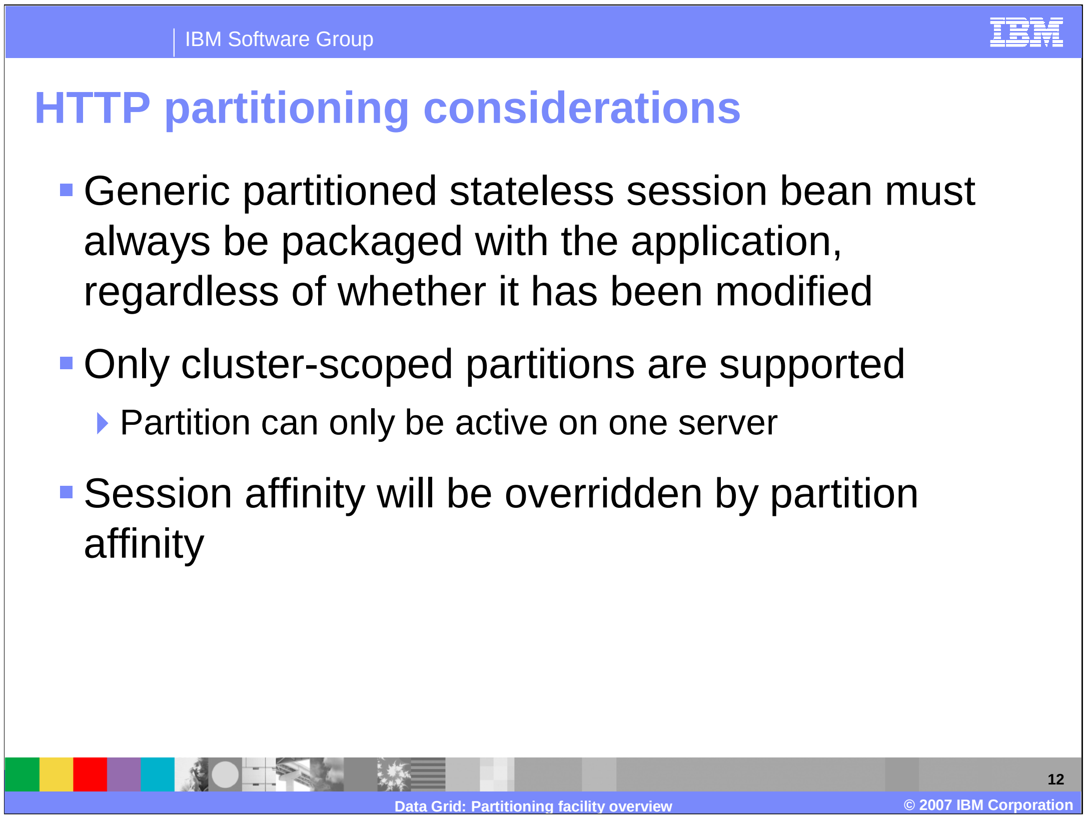
HTTP partitioning

- The on-demand router can integrate with the partitioning facility to implement partitioning for HTTP requests
- A partitioning facility module runs inside the on demand router
  - Keeps an updated list of partitions
  - Gets current partition-to-server mapping from partitioning facility
  - Maps requests to partitions using regular expressions
- Partitions can be specified programmatically or by deploying partition.xml within the application

11

Data Grid: Partitioning facility overview © 2007 IBM Corporation

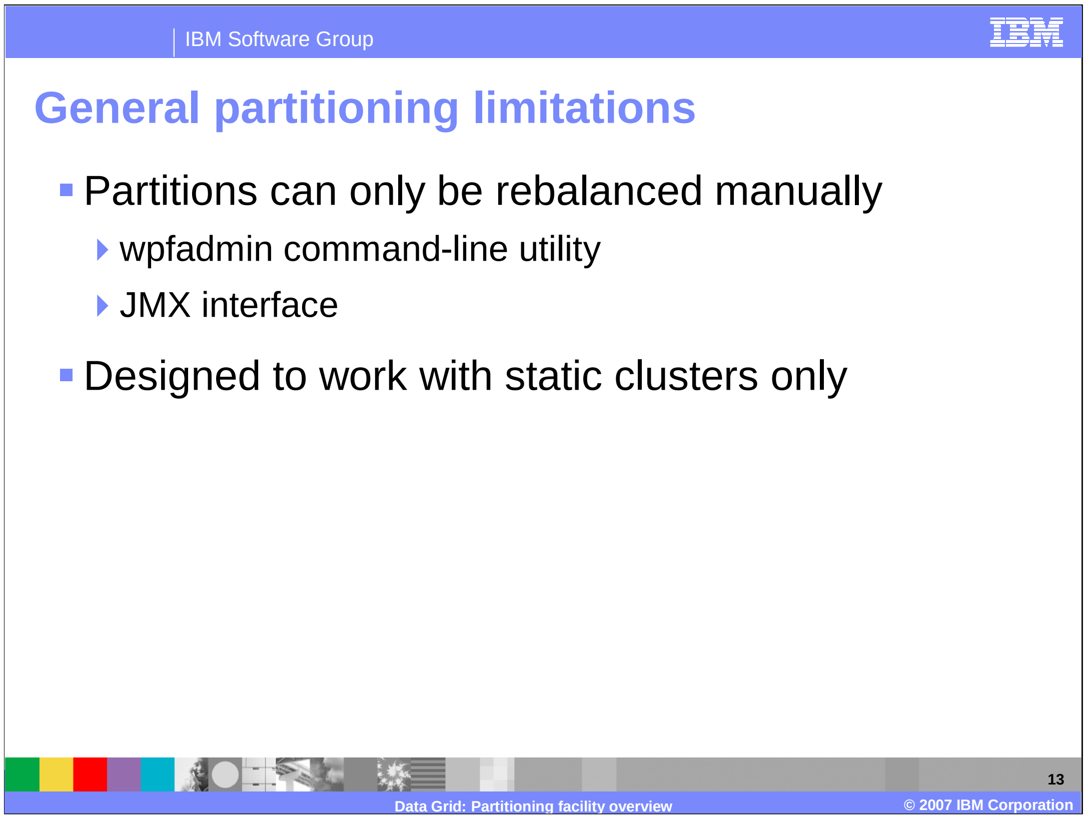
WebSphere Extended Deployment also provides support for partitioning of applications that are accessed over HTTP, which is particularly useful when Web module and EJB module are collocated. This means that HTTP requests are predictably routed to a single endpoint, just as IIOP requests are. To accomplish HTTP partitioning, the on demand router interacts with the partitioning facility to route requests to unique partitions based on regular expressions specified by a partitioned stateless session bean or by a partition.xml file.



HTTP partitioning considerations

- Generic partitioned stateless session bean must always be packaged with the application, regardless of whether it has been modified
- Only cluster-scoped partitions are supported
  - Partition can only be active on one server
- Session affinity will be overridden by partition affinity

A partitioned HTTP application must always contain a partitioned stateless session bean, even if the partition mappings are defined in a partition.xml file, because the existence of this bean is what identifies an application as being partitioned. HTTP partitions can exist only on one server within a cluster, so node-scoped partitioning, which is an option for EJB partitioning, is not available for HTTP partitioning. Also, HTTP partition definitions take precedence over normal HTTP session affinity.


IBM Software Group

General partitioning limitations

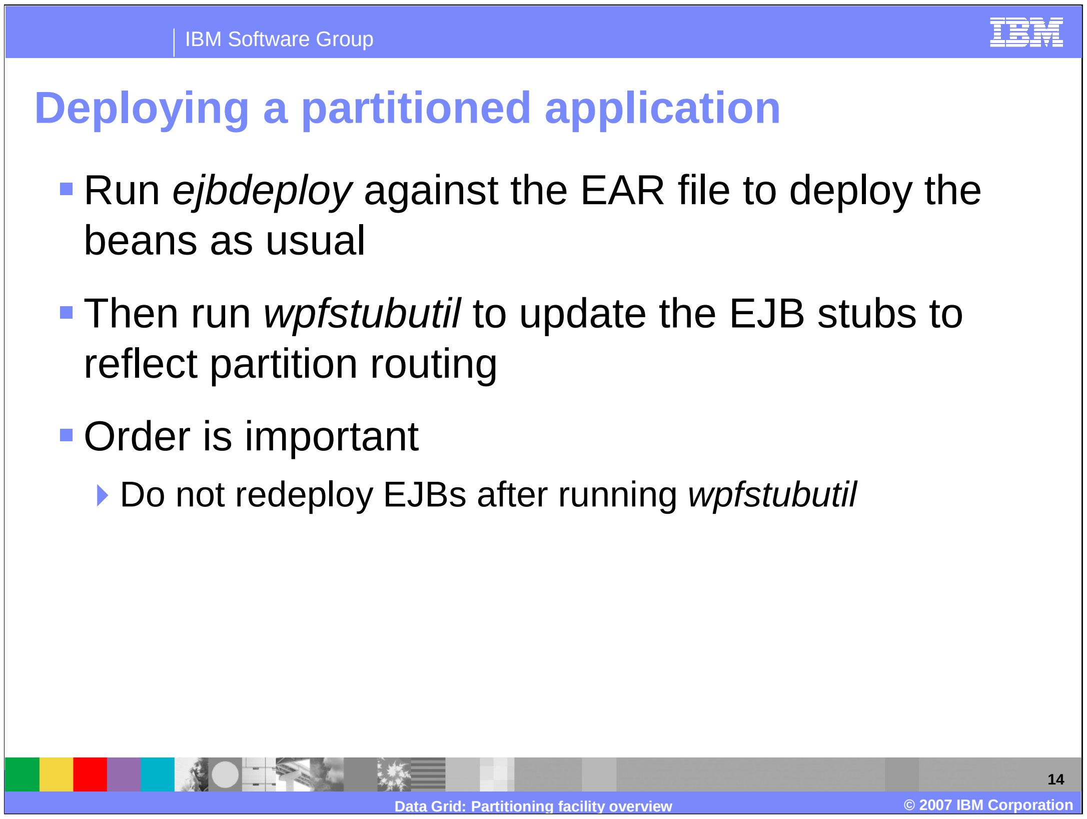
- Partitions can only be rebalanced manually
  - wpfadmin command-line utility
  - JMX interface
- Designed to work with static clusters only

13

Data Grid: Partitioning facility overview

© 2007 IBM Corporation

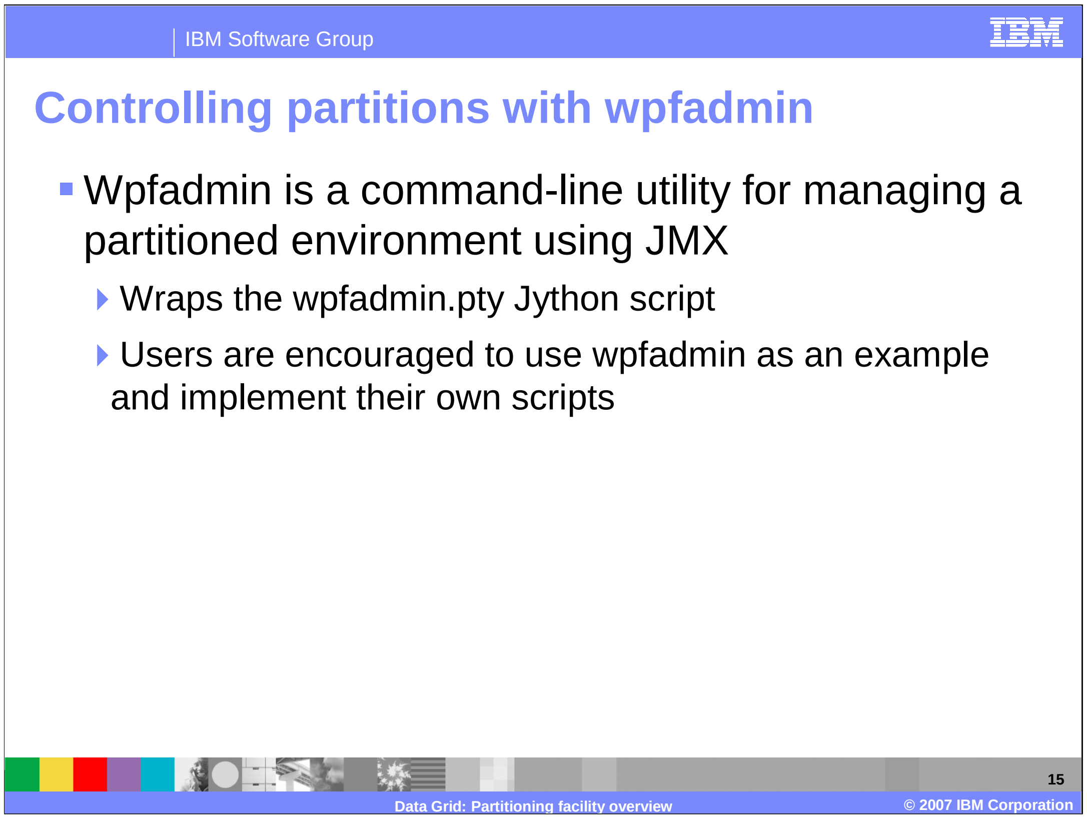
While the partitioning facility will reallocate partitions when a server stops, it does not automatically rebalance partitions as new server instances start. As a result of this, partitioning facility is not designed to work with dynamic clusters.



Deploying a partitioned application

- Run *ejbdeploy* against the EAR file to deploy the beans as usual
- Then run *wpfstubutil* to update the EJB stubs to reflect partition routing
- Order is important
  - Do not redeploy EJBs after running *wpfstubutil*

An extra step is required to deploy a partitioned application. Use the 'ejbdeploy' tool to deploy the enterprise beans, as you would for any other application. Then run the 'wpfstubutil' command, which modifies the EJB stubs to reflect the partitioned routing scheme. It is important not to redeploy the EJBs after calling 'wpfstubutil', because the partitioning-specific changes will be undone, requiring you to run 'wpfstubutil' again.



Controlling partitions with wpfadmin

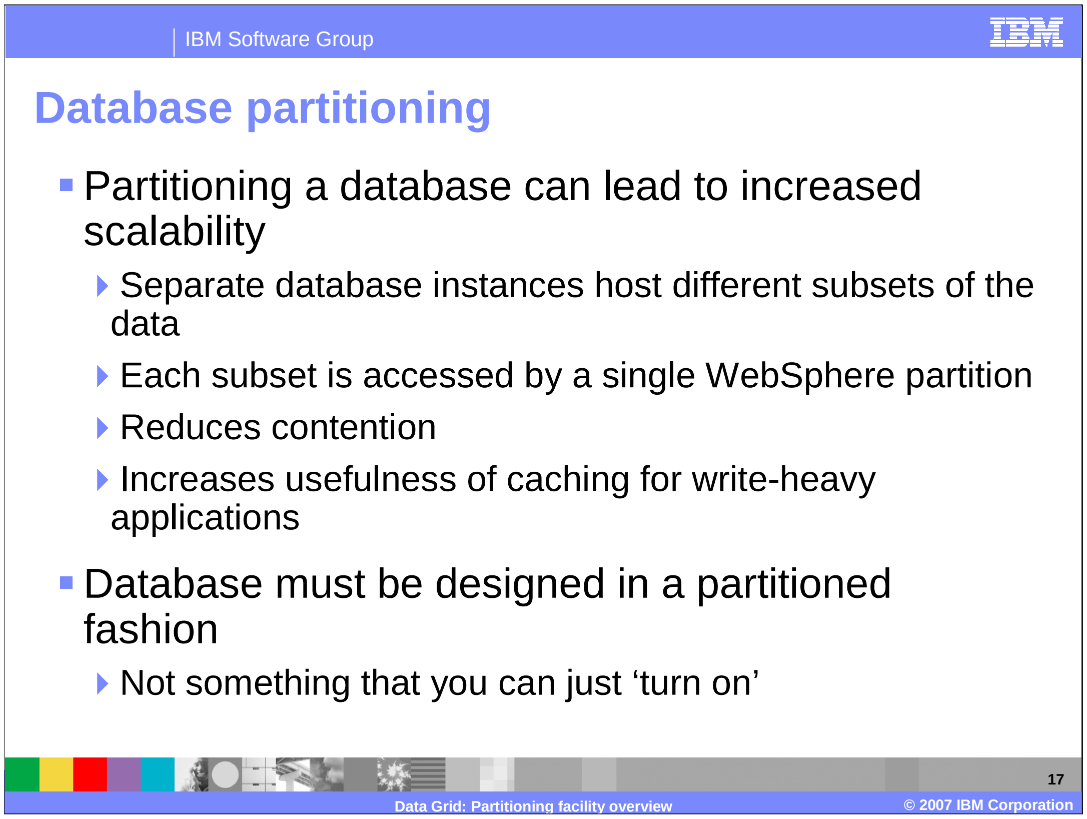
- Wpfadmin is a command-line utility for managing a partitioned environment using JMX
  - Wraps the wpfadmin.pty Jython script
  - Users are encouraged to use wpfadmin as an example and implement their own scripts

WebSphere Extended Deployment provides a command line tool, ‘wpfadmin’, for managing a partitioned application environment. Wpfadmin allows you to view the location of partitions within your cluster, to move partitions from one server to another, and to configure the high availability manager, among other things. Wpfadmin is an interface to a Jython script, wpfadmin.pty, that is well documented and makes a useful example for implementing your own partitioning facility management scripts.

Section

Database partitioning

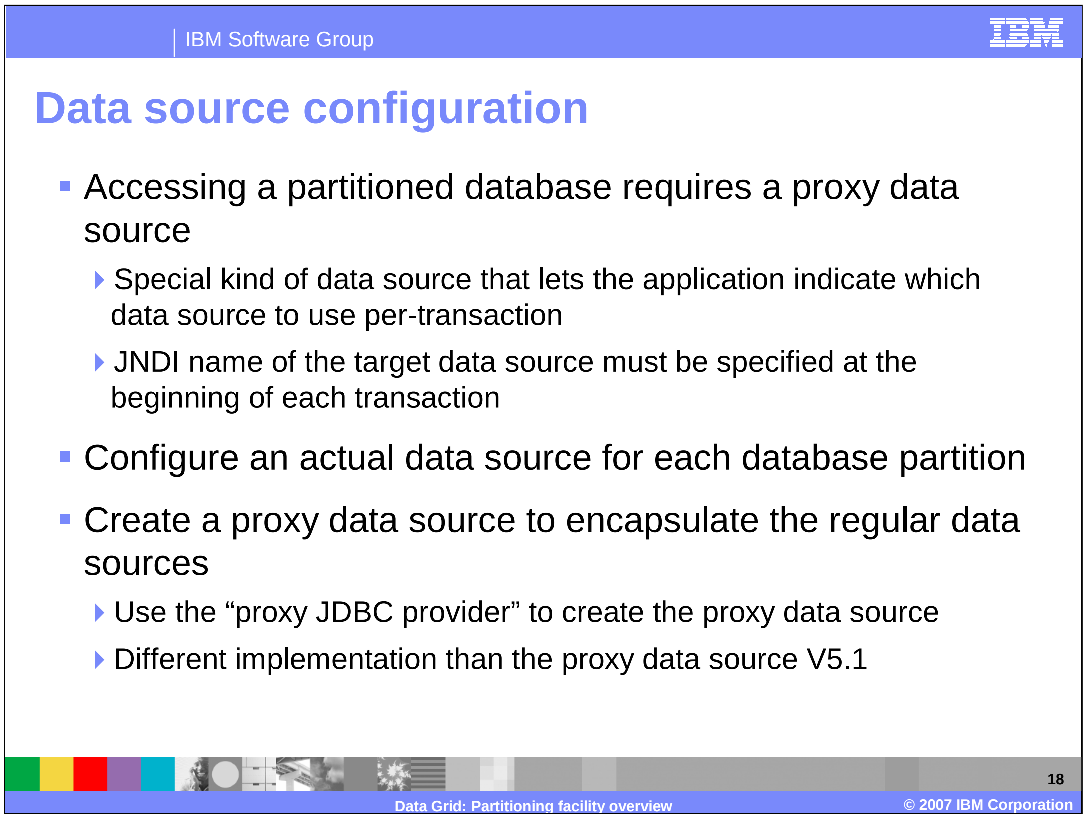
This section will discuss database partitioning.



Database partitioning

- Partitioning a database can lead to increased scalability
  - Separate database instances host different subsets of the data
  - Each subset is accessed by a single WebSphere partition
  - Reduces contention
  - Increases usefulness of caching for write-heavy applications
- Database must be designed in a partitioned fashion
  - Not something that you can just 'turn on'

In addition to partitioning your application, partitioning the database that your application accesses can drastically increase scalability. Database partitioning involves creating a separate database instance for the subset of data that will be accessed by each partition. This gives each partition exclusive access to the data that it will be using, reducing database contention and giving the application the freedom to more aggressively cache values in memory. It also gives you the ability to easily scale the database across multiple servers, rather than implementing a database clustering solution. Database partitioning requires careful planning and consideration at the time of application design; it is not a feature for which there is a simple on-off switch


IBM Software Group

Data source configuration

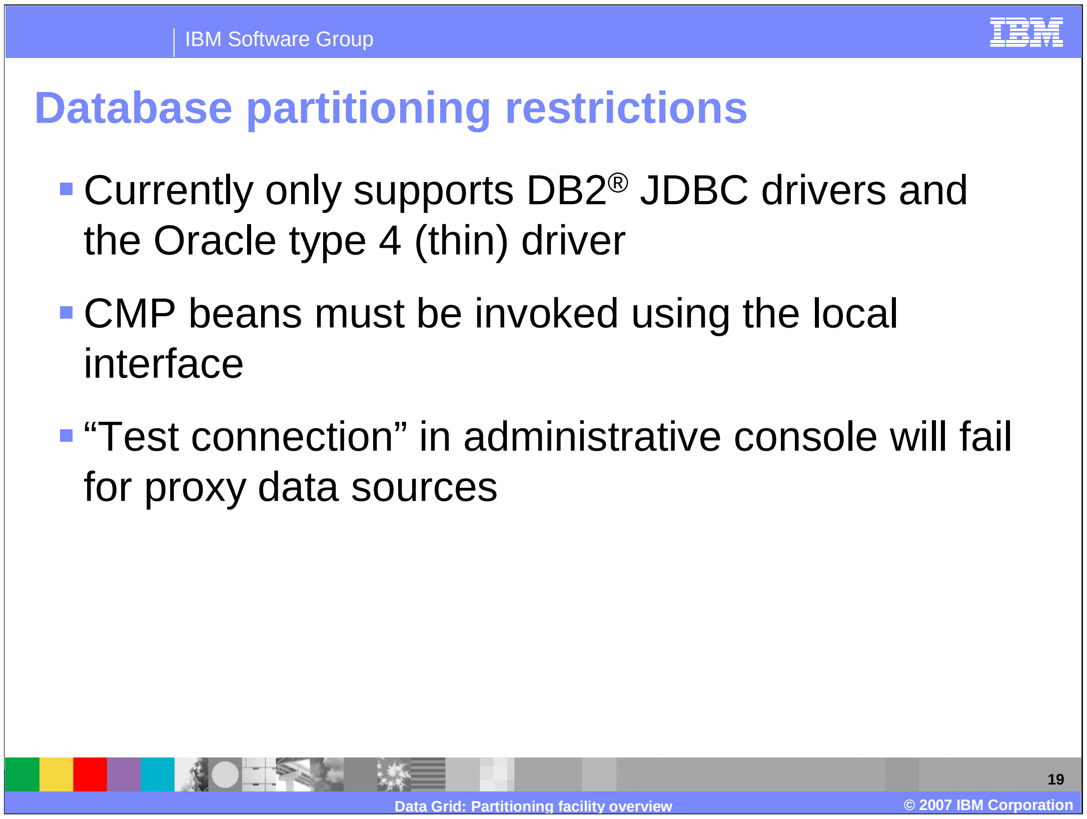
- Accessing a partitioned database requires a proxy data source
  - Special kind of data source that lets the application indicate which data source to use per-transaction
  - JNDI name of the target data source must be specified at the beginning of each transaction
- Configure an actual data source for each database partition
- Create a proxy data source to encapsulate the regular data sources
  - Use the “proxy JDBC provider” to create the proxy data source
  - Different implementation than the proxy data source V5.1

18

Data Grid: Partitioning facility overview

© 2007 IBM Corporation

WebSphere Extended Deployment enables applications to access partitioned databases by utilizing a special Data Source, called a proxy data source. A proxy data source allows an application to specify the name of another data source to use for each database transaction. The application therefore has the ability to dynamically access different database instances based on the partition context of each request.



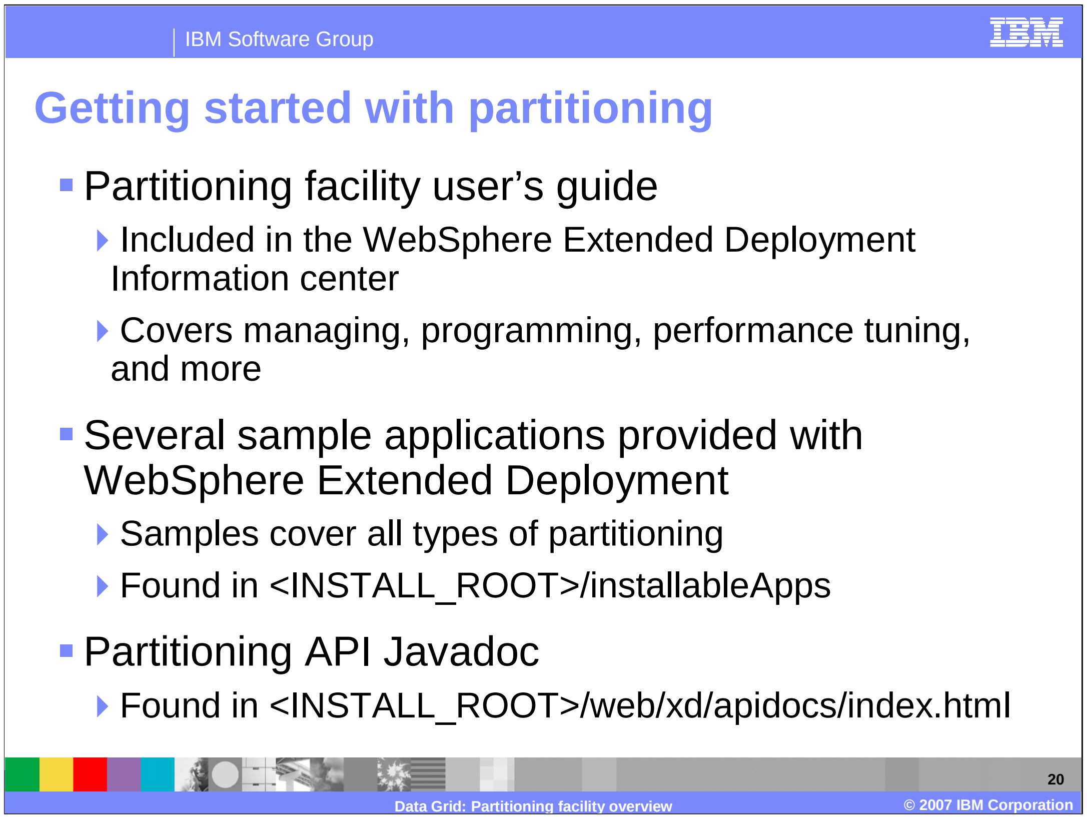
Database partitioning restrictions

- Currently only supports DB2® JDBC drivers and the Oracle type 4 (thin) driver
- CMP beans must be invoked using the local interface
- “Test connection” in administrative console will fail for proxy data sources

WebSphere Extended Deployment currently only supports database partitioning when using the DB2 JDBC drivers or the Oracle Type 4 JDBC driver.

When using a proxy data source, application developers must use a session EJB as the session facade. The session EJB uses the container managed persistence enterprise beans’ local interfaces to interact with container managed persistence EJBs.

Also, the “test connection” feature of the administrative console does not support proxy data sources.


IBM Software Group

Getting started with partitioning

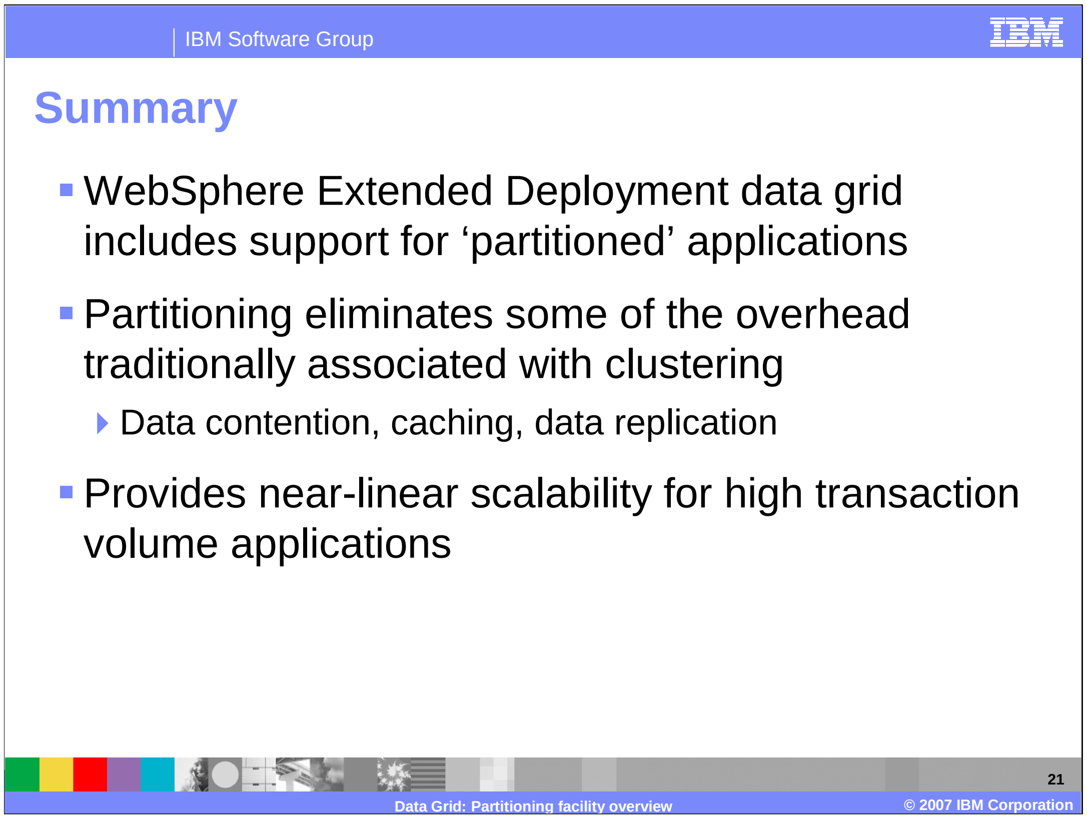
- Partitioning facility user's guide
  - Included in the WebSphere Extended Deployment Information center
  - Covers managing, programming, performance tuning, and more
- Several sample applications provided with WebSphere Extended Deployment
  - Samples cover all types of partitioning
  - Found in <INSTALL_ROOT>/installableApps
- Partitioning API Javadoc
  - Found in <INSTALL_ROOT>/web/xd/apidocs/index.html

20

Data Grid: Partitioning facility overview

© 2007 IBM Corporation

Several helpful resources are available to help you get started with the partitioning facility. The partitioning facility user's guide is included in the WebSphere Extended Deployment Information Center, and thoroughly covers developing, managing, and tuning partitioned applications. You can also find several samples of partitioned applications in the 'installableApps' directory after installing WebSphere Extended Deployment data grid. Finally, the Javadoc installed with the product is the official documentation for the partitioning API.



IBM Software Group

Summary

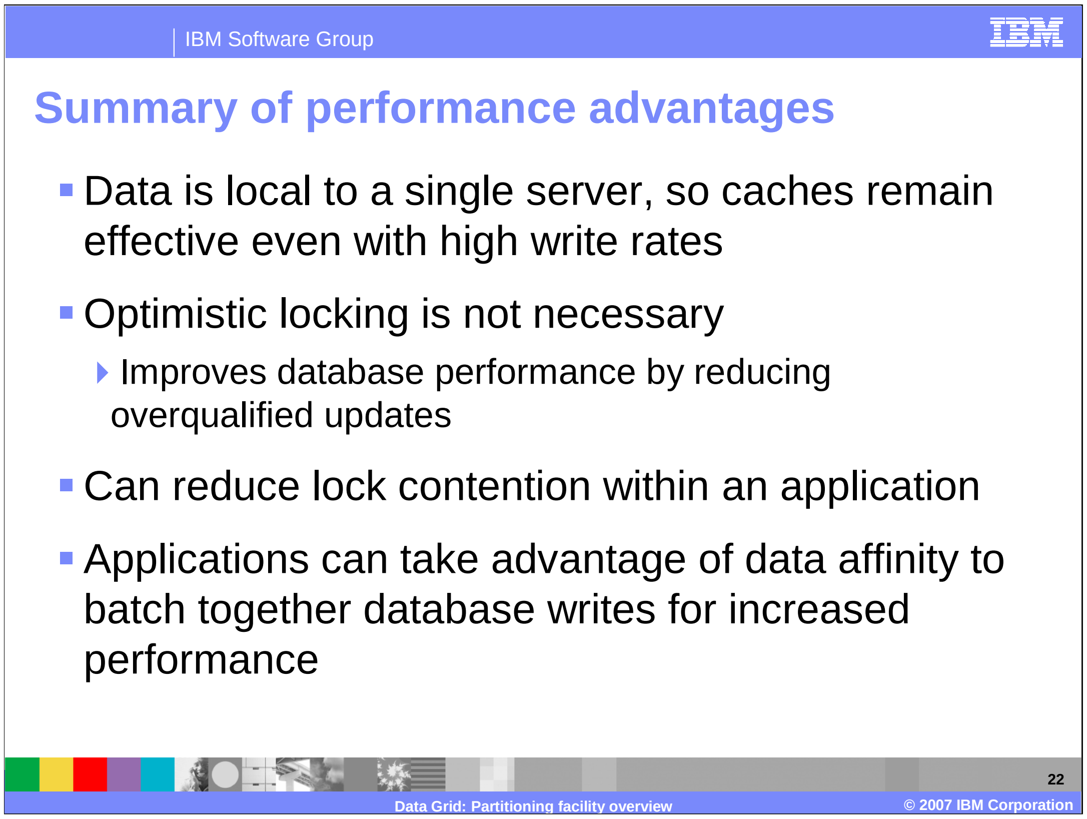
- WebSphere Extended Deployment data grid includes support for ‘partitioned’ applications
- Partitioning eliminates some of the overhead traditionally associated with clustering
  - Data contention, caching, data replication
- Provides near-linear scalability for high transaction volume applications

21

Data Grid: Partitioning facility overview

© 2007 IBM Corporation

In summary, the partitioning facility in WebSphere Extended Deployment Data Grid is a programming framework and runtime infrastructure that enables you to build high-performance J2EE applications that can scale efficiently on distributed hardware. Partitioning addresses the drawbacks typically associated with large-scale clustering, such as data contention, distributed caching, and data replication.


IBM Software Group

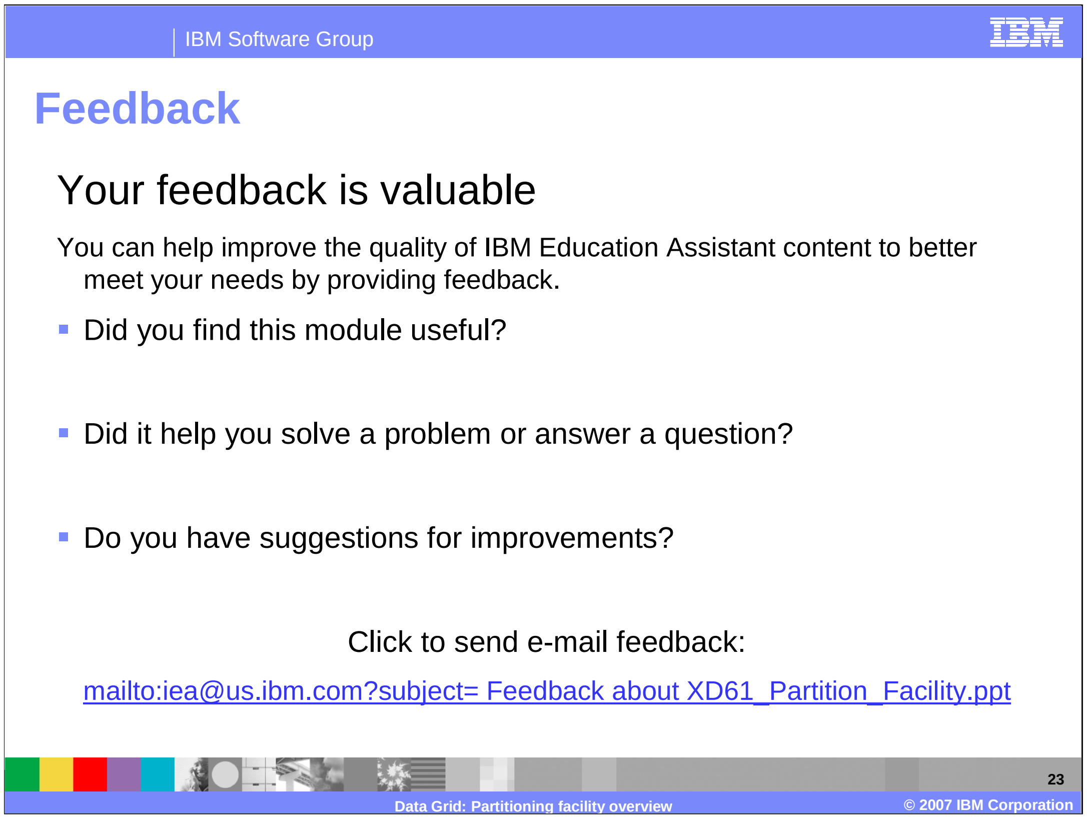
Summary of performance advantages

- Data is local to a single server, so caches remain effective even with high write rates
- Optimistic locking is not necessary
  - Improves database performance by reducing overqualified updates
- Can reduce lock contention within an application
- Applications can take advantage of data affinity to batch together database writes for increased performance

22

Data Grid: Partitioning facility overview © 2007 IBM Corporation

Many potential performance advantages are inherent in the partitioned application model. Partitioned applications can cache data very aggressively, and these caches are effective even with high write rates. The need for overqualified updates introduced by optimistic locking is reduced when using partitioning, since each partition has exclusive access to a database instance. Partitioning can also help increase parallelism within an application, because lock contention within the application server itself can also be reduced. Finally, applications can be designed to take advantage of the fact that they exclusively handle requests for certain data sets by performing batched database writes to decrease load on the database.



Feedback

Your feedback is valuable

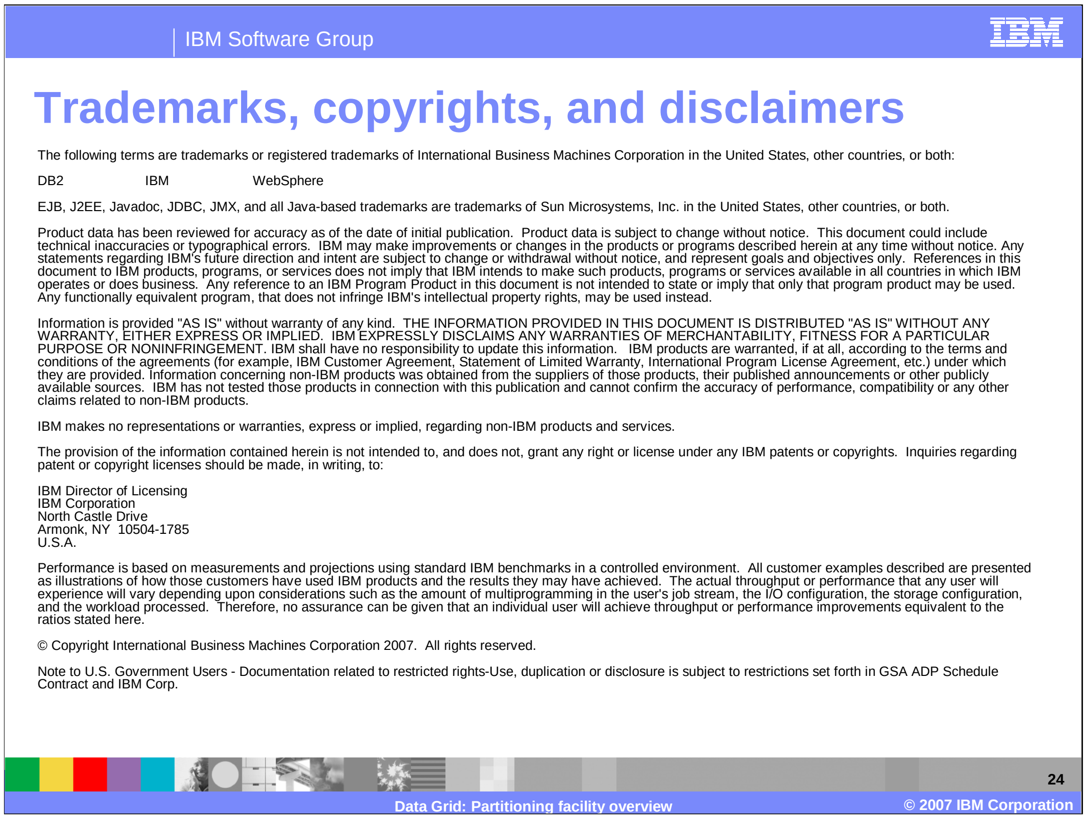
You can help improve the quality of IBM Education Assistant content to better meet your needs by providing feedback.

- Did you find this module useful?
- Did it help you solve a problem or answer a question?
- Do you have suggestions for improvements?

Click to send e-mail feedback:

mailto:iea@us.ibm.com?subject= Feedback about XD61_Partition_Facility.ppt

You can help improve the quality of IBM Education Assistant content by providing feedback.



Trademarks, copyrights, and disclaimers