Lesson 1

Clustering tables and indexes

One advantage of a relational database is that users do not need to know anything about the physical structure of the database.   
They can access all the data by simply using Structured Query Language (SQL). However, underneath the *logical structures*manipulated by SQL, the data in your Oracle database is still stored on a disk. And, as with all data on disk, the performance of data retrieval is dependent to some extent on the physical movement of the disk head over the spinning disk. If you can minimize this disk head movement, you minimize the single largest bottleneck in data retrieval. Oracle allows you to create *clustered* tables and indexes, whose primary purpose is to minimize the inherent delays caused by disk head movement. This module covers all you need to know about clusters, including:

1. The advantages of clustering
2. How to create and size a cluster
3. How to create a cluster key
4. The purpose of a hash cluster
5. How to create a hash cluster
6. How to drop a cluster

**Clusters**

*Clustering* is a method of storing tables that are intimately related and often joined together into the same area on disk. For example, instead of the BOOKSHELF table being in one section of the disk and the BOOKSHELF\_AUTHOR table being somewhere else, their rows could be interleaved together in a single area, called a cluster. The cluster key is the column or columns by which the tables are usually joined in a query (for example, Title for the BOOKSHELF and BOOKSHELF\_ AUTHOR tables). To cluster tables, you must own the tables you are going to cluster together. The following is the basic format of the create cluster command:

create cluster cluster

(column datatype [,column datatype]. . .) [other options];

The cluster name follows the table-naming conventions, and column datatype is the name and datatype you will use as the cluster key. The column name may be the same as one of the columns of a table you will put in this cluster, or it may be any other valid name. Here is an example:

create cluster BOOKandAUTHOR (Col1 VARCHAR2(100));

This creates a cluster (a space is set aside, as it would be for a table) with nothing in it. The use of Col1 for the cluster key is irrelevant; you will never use it again. However, its definition should match the primary key of the table to be added. Next, tables are created to be included in this cluster:

create table BOOKSHELF

(Title VARCHAR2(100) primary key,

Publisher VARCHAR2(20),

CategoryName VARCHAR2(20),

Rating VARCHAR2(2),

constraint CATFK foreign key (CategoryName)

references CATEGORY(CategoryName)

)

cluster BOOKandAUTHOR (Title);

Prior to inserting rows into BOOKSHELF, you must create a cluster index:

create index BOOKandAUTHORndx

on cluster BOOKandAUTHOR;

Recall that the presence of a cluster clause here precludes the use of a tablespace or storage clause. Note how this structure differs from a standard create table statement:

create table BOOKSHELF

(Title VARCHAR2(100) primary key,

Publisher VARCHAR2(20),

CategoryName VARCHAR2(20),

Rating VARCHAR2(2),

constraint CATFK foreign key (CategoryName)

references CATEGORY(CategoryName)

);

Although clusters are not appropriate for all types of data, in the right situation a cluster can provide a significant increase in performance.

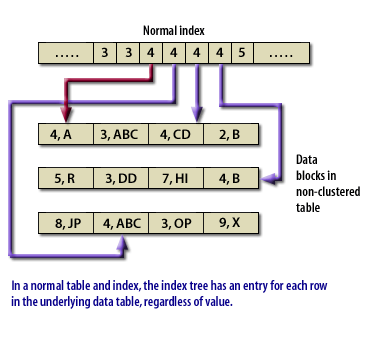
In the next lesson you will learn when to use clustered tables.

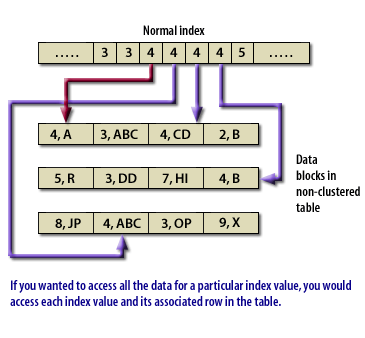
|  |  |
| --- | --- |
| Lesson 2 | Advantages of clusters |
| Objective | Know when to use clustered tables. |

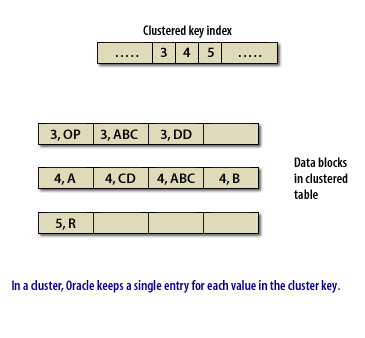
In Theory, each data object within an Oracle database is a totally separate entity.   
In practice, certain tables are frequently used together, especially when using a *normalized database design*.   
A cluster is a way of organizing data to leverage the related nature of data stored in different locations.

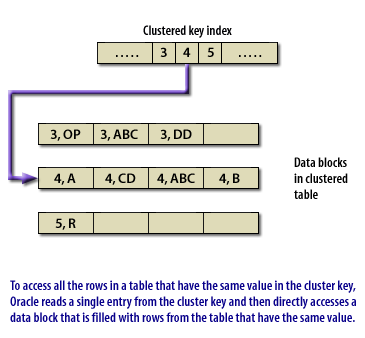
What is a cluster?

Simply put, a cluster ties data values to disk location.   
A *cluster key* is used to group data together. All rows of all tables with the same value of the cluster key are stored in the same data block.   
The following Slide Show shows the difference between a normal table and a clustered table.









[Prev](javascript:firstreel.navigate('back')) [Next](javascript:firstreel.navigate('forth'))

[Clustered Versus Non-Clustered](http://www.relationaldbdesign.com/extended-database-features/module3/clustered-versus-non-clustered.php)   
Advantages of a cluster

Since related data is stored together, the related data can be accessed with fewer data block reads. With a cluster, Oracle reads the cluster key, which directly points to the disk area that contains the data for that value of the key. The cluster delivers two advantages:

1. First of all, a query can retrieve all the related data with usually no more than 2 logical reads--one to get the cluster key and another to retrieve a data block that contains only the relevant data. For instance, if an employee table is clustered on the department number, a query for employees in a department would first retrieve the relevant cluster key for the department and then the employee rows for that department. If an order header table and an order detail table were clustered together, one read would retrieve the order cluster key and one more read would retrieve both the order header and detail rows for that order.
2. The second advantage is that the value for the cluster key is only stored once, and this reduces the amount of space required for storage.

When to use a cluster

The ideal places to use a cluster are when you have a group of tables that are frequently queried together, or when you have a single table that is frequently accessed by an index value. For example, you might cluster an order header table and an order detail table together. If you frequently accessed employees by department, you may want to cluster an employee table by department.

When not to use a cluster

In some situations, a cluster is definitely not appropriate. You should not use a cluster in the following situations:

1. If the value for a cluster key is updated frequently.
2. If the data for the values of the cluster key takes up more than one or two Oracle data blocks.
3. If you frequently require full table scans on the clustered data.

For an explanation of why clustering is not appropriate in these cases, check this   
[Oracle Clustering Advantages](http://www.relationaldbdesign.com/extended-database-features/module3/oracle-clustering-advantages.php).

**When is clustering bad?**

Clustering does not work well for all tables. For instance, clustering is inadvisable in the following three scenarios:

1. Frequent updates of the cluster key-Remember that data is stored in a data block based on the value of the cluster key. If a user updates that value, the actual data row will have to be moved to a different data block, which is more resource intensive than a simple in-place update of a non-clustered row.
2. Data takes up more than one or two blocks-The big advantage of clusters comes from the ability to read a data block that contains all the rows for a particular value for the cluster key. However, the cluster is set up so that the cluster key points to the first data block for the cluster. If there are multiple data blocks for a value, and you are trying to retrieve a specific row that is in a later block, you will actually increase the number of database reads with a cluster. This is because Oracle has to read through the values for the cluster from the beginning.
3. Frequent table scans required-In a cluster, the data is stored according to a value in the cluster key. Because you may have a cluster that includes data from more than one table, the overall size of the cluster will be the combined size of the data in all the tables in the cluster. This combination means that a table scan of a single table in the cluster will take longer, and possiblymuch longer, than scanning the data for a single table.

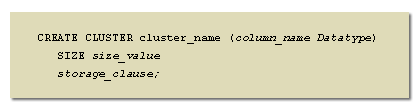
The next lesson shows how to create a cluster.

|  |  |
| --- | --- |
| Lesson 3 | Creating a cluster |
| Objective | Deciding which table or tables to cluster |

The most important step in creating a cluster is deciding which table or tables to cluster, and how. Although a clustered table can deliver a terrific performance improvement in the right circumstances, a poorly chosen cluster can decrease performance. Once you have properly selected the table or tables you wish to cluster, you must go through a three-step process to create this database structure.

## Create the cluster

The following Tooltip illustrates how to create a cluster using SQL:



[Oracle Cluster Advantages](http://www.relationaldbdesign.com/extended-database-features/module3/create-cluster.php)   
  
Add tables to the cluster

Once you create a cluster, you then create the table or tables that the cluster will contain.   
The syntax for creating tables that are a part of a cluster is exactly the same syntax that is used for non-clustered tables, with one exception. The final clause in the CREATE TABLE statement is:

CLUSTER cluster\_name (column\_name)

The cluster\_name is the same name that was given to the cluster in the CREATE CLUSTER command. The column\_name is a list of columns in the table being created that match up with the columns in the already created cluster.

Create the cluster key

The final step is to create a cluster index. You will learn to do this in Lesson 5 of this module.

Cluster Example

The following simplified code is an example of creating a cluster and the tables it will contain:

CREATE CLUSTER orders (order\_id NUMBER) SIZE 512 K;

CREATE TABLE order\_header( order\_number NUMBER,

customer\_number NUMBER)

CLUSTER orders (order\_number);

CREATE TABLE order\_detail(

order\_number NUMBER,

detail\_line VARCHAR2(100))

CLUSTER orders (order\_number);

The next lesson explains how to size a cluster properly.

|  |  |
| --- | --- |
| Lesson 4 | Sizing a cluster |
| Objective | Properly size a cluster. |

In a non-clustered table, the size of the data to be contained in the table is not a great concern.   
A DBA always must be sure to have enough disk space to store data, but management of the underlying storage is handled by your Oracle database.   
When you use a cluster, however, you are explicitly linking data, by value, with the way it is stored on disk.   
Because of this, you must specify a size for the data associated with a particular value of the cluster key.

How do you size a cluster?

The CREATE CLUSTER command has a required SIZE parameter.   
You can specify this size either in K(ilobytes) or M(egabytes). When you create a cluster, each time you add a value for the cluster key, Oracle allocates the number of data blocks required by the value in the SIZE clause. The size specified in the CREATE CLUSTER should be large enough to store all of the data for all of the rows associated with the cluster value and the cluster value itself.

Improperly size your cluster?

If you size your cluster too small, the data for the cluster value will extend beyond the pre-allocated space, and the additional chaining will reduce the overall performance advantages of the cluster. If you specify too large a size for your cluster, you will end up wasting disk space, which can also contribute to reduced disk and database performance.   
The next lesson shows how to create a cluster index.

Sizing Cluster - Exercise

Click the Exercise link below to practice granting access to database objects.   
[Sizing Cluster - Exercise](http://www.relationaldbdesign.com/extended-database-features/module3/sizing-cluster-exercise.php)

Sizing an Oracle Cluster - Exercise

Top of Form

Creating a cluster  
Objective:Create a cluster in the COIN database  
Exercise scoring

This exercise is worth a total of 20 points, 10 points for each SQL statement that you complete correctly. You may not receive full credit for the answer if your syntax is only partially correct.   
Once you complete your answer, submit your answer.

Background/overview

In previous courses, you created tables for the COIN database.

Two of these tables have a master-detail relationship, the LOT table and the COINS\_IN\_LOT table, so you might want to consider creating a cluster with those two tables. The following is a simplified form of the SQL statements that were used to create these two tables.

CREATE TABLE LOT (auction\_id NUMBER, lot\_id NUMBER,

min\_prince NUMBER(11,2), winning\_bid NUMBER(11,2),

winning\_bidder NUMBER, CONSTRAINT pk\_lot PRIMARY KEY

(auction\_id, lot\_id));

CREATE TABLE COINS\_IN\_LOT (auction\_id NUMBER, lot\_id NUMBER,

coin\_id NUMBER, CONSTRAINT pk\_coins\_in\_lot PRIMARY KEY

(auction\_id, lot\_id, coin\_id));

As mentioned above, you must consider multiple variables before you create a cluster, such as if the data in the tables will be accessed through a full table scan. Depending on the usage of these tables in your application, you may or may not want to make them a cluster in real life.

Download files

You can download a correct version of the script for this exercise from the Resources page.

Instructions

You will write SQL statements to create a cluster that contains the LOT and COINS\_IN\_LOT tables.

Hints

You need to go through two separate steps to create the cluster and the tables in the cluster.  
You should give the cluster a size of 512 K. The cluster key for this cluster will contain two columns.

Submitting your exercise

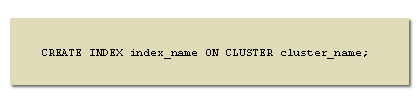
Enter your answer in the text area below and click the **Submit** button when you are ready to submit your answer.

Bottom of Form

|  |  |
| --- | --- |
| Lesson 5 | Creating a cluster key |
| Objective | Creating a cluster key |

The third step in creating a cluster is creating a cluster index. A cluster index is an index structure that contains the values of the cluster key, the column or columns that link the tables in the cluster together. The cluster index is more than just a way to optimize retrieval. If you have a cluster, you *must* have a cluster index because of the way data is accessed through a cluster. The order for getting the data is to go to the cluster index, find the cluster value, and then go directly to the first data block on the disk where all the rows with that value reside. If there is no cluster index, this chain is broken. If you drop a cluster index, you effectively remove the data in the cluster from the database because users can not get to the data without an index. In addition, a cluster index differs from a standard index in that a cluster index contains a single entry for each value in the index, not for each row in the underlying table.

Syntax  
The SQL syntax for creating a cluster index is:



[Create Index Cluster](http://www.relationaldbdesign.com/extended-database-features/module3/create-index-cluster.php) 

You do not have to list any of the columns in the cluster key, because they have already been identified in the CREATE CLUSTERcommand. You can have additional clauses on the CREATE INDEX statement for a cluster to specify additional attributes like the tablespace for the index, just as you would for a normal index.   
To create a cluster index for a cluster named *coin\_lot*, you would use the following SQL statement:   
The next lesson introduces a special type of cluster called a hash cluster.

CREATE INDEX coin\_lot\_idx ON CLUSTER coin\_lot;

Creating Cluster Key - Quiz

Click the Quiz link below to answer a few questions about clusters and cluster indexes.   
[Creating Cluster Key - Quiz](http://www.relationaldbdesign.com/extended-database-features/module3/creating-cluster-key-quiz.php)

Creating Cluster Key - Quiz

Each question is worth one point. Select the best answer or answers for each question.

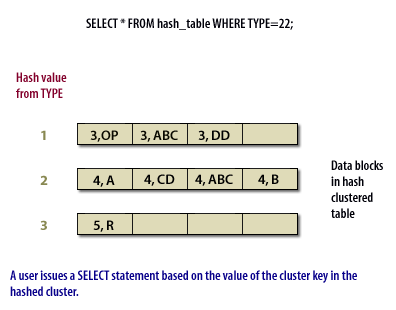
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Top of Form   |  |  |  | | --- | --- | --- | | 1. | Which of the following steps is not necessary to create a working cluster? Please select the best answer. | | |  | A. | Create the cluster | |  | B. | Create the tables that will be in the cluster | |  | C. | Create the cluster index | |  | D. | Create a foreign key constraint between the tables in the cluster |      |  |  |  | | --- | --- | --- | | 2. | If you drop a cluster index, what is the effect on accessing the data in the cluster? Please select the best answer. | | |  | A. | Data access for the cluster will be slower | |  | B. | Data access for the cluster will be faster | |  | C. | Data access for the cluster will not be possible | |  | D. | Data access for the cluster will not be affected |  |  |  |  | | --- | --- | --- | | 3. | Which types of data might be appropriate for use in a cluster? Please select all the correct answers. | | |  | A. | All data is appropriate for a cluster | |  | B. | Tables with a master-detail relationship | |  | C. | Tables where the cluster key is frequently updated | |  | D. | Data usually retrieved by selecting a value from a cluster key |     Bottom of Form |

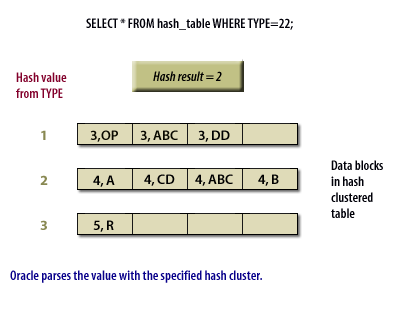
|  |  |
| --- | --- |
| Lesson 6 | Hash clusters |
| Objective | Know When to use Oracle Hash Cluster |

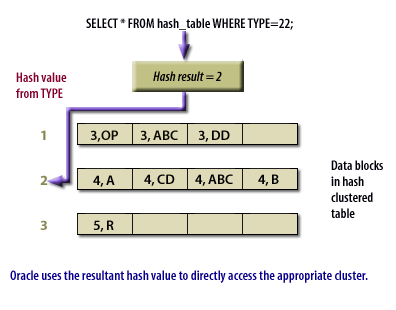
There is another type of clustering available in your Oracle database that is called a hash cluster.   
A hash cluster is similar in most respects to a standard cluster. The data in a hash cluster is grouped according to a value, which is stored in a cluster index.

How is a hash cluster difference?

The big difference between a hash cluster and a normal cluster is the way the data is accessed. In a regular cluster, Oracle uses the value of the cluster key to access the data in the cluster. In a hash cluster, Oracle uses the value of a hashing function to access the data in the cluster.   
The following slide show illustrates a hash cluster in action:







[Prev](javascript:firstreel.navigate('back')) [Next](javascript:firstreel.navigate('forth'))

[Hash Table Cluster](http://www.relationaldbdesign.com/extended-database-features/module3/hash-table-cluster.php)

The hashing function is an algorithm that acts on the values in the cluster key. Because of this, all the values in a cluster key for a hash cluster must be numeric. Oracle can either use its standard hashing algorithm, or you can assign a specific hashing function for a hash cluster. A hash cluster differs from a normal cluster in that the hash value is not stored in the cluster itself. In addition, although a cluster index is required for a regular cluster, you *cannot* create a cluster index on a hash cluster. Because there is no cluster index, the number of I/O operations is cut in half.

When should you use a hash cluster?

A cluster contains all the data for a specific value of the cluster key in a limited number of data blocks. You may have data that is appropriate for a cluster, but whose cluster key has an unequal distribution of values. This would result in the data for some key values being much larger than others, which would in turn force you to size the cluster to accommodate the largest sets of rows, which would waste space for the smaller sets of rows. In this case, you might use a hash cluster to force a more even distribution of rows and still get the advantages of a cluster.   
The next lesson explores how to create and size a hash cluster.

|  |  |
| --- | --- |
| Lesson 7 | Creating a hash cluster |
| Objective | Create an Oracle Hash cluster. |

Just as a hash cluster is different from a normal cluster, creating a hash cluster is different from creating a standard cluster.   
When Oracle creates a hash cluster table, it immediately allocates *all* the space that will be required by all the data in the hash cluster. Oracle determines the total amount of space for the hash cluster by using the SIZE parameter, which you learned about earlier, and another parameter called HASHKEYS. The value for the HASHKEYS parameter limits the total number of unique values for the result of the hashing function. If you specify a HASHKEYS value of 100, there will be no more than 100 different values for the result of the hashing function. If there are more than 100 possible results, Oracle will still only create 100 different areas to correspond to 100 different values, so some collisions, where multiple values of the hash function are stored together, will occur.

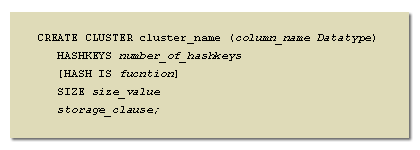
Types of hashing

There are three different types of hash functions you can choose for a hash cluster. You can specify:

1. Oracle's internal hashing function.
2. The value of the cluster key as the result of the hash function. If the value of the cluster key is greater than the value specified for HASHKEYS, the value is divided by the HASHKEYS value and the remainder is used as the hash key.
3. Any SQL function.

Syntax

In order to create a hash cluster, you use the syntax shown in the following Tooltip:



[Oracle Create Hash Cluster](http://www.relationaldbdesign.com/extended-database-features/module3/oracle-create-hash-cluster.php)  
Hash Cluster Example

If you wanted to create a hash cluster for the LOT table, base it on value of the LOT\_ID column, allow 100 values, and use the MOD SQL function to return a remainder from dividing the LOT\_ID column by 100, you would use the following command:

CREATE CLUSTER lot\_cluster (lot\_id NUMBER)

SIZE 1 M

HASHKEYS 100

HASH IS MOD(lot\_id, 100);

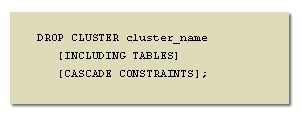
The next lesson explains how to delete clusters.

|  |
| --- |
| Deleting a cluster |
| Objective | Drop a cluster. |

A cluster is a database object and like other database objects, you can delete is from the database.

DROP CLUSTER

To drop a cluster, you use the syntax illustrated in the following display:



[Drop Oracle Cluster](http://www.relationaldbdesign.com/extended-database-features/module3/drop-oracle-cluster.php) 

When you drop a cluster, the cluster index, if one exists, is also dropped. You can drop a table from a cluster that contains multiple tables by using the DROP TABLE command. However, the result of this action will be that Oracle individually deletes each row of the table. There is no way to uncluster a table, since the cluster actually controls the physical placement of the table on the disk. If you want to change a clustered table to an unclustered table, you must first unload the data from the table, drop the cluster, create the table again without a CLUSTER clause, and reload the data back into the table.

Oracle Cluster Example

If you wanted to drop the existing lot\_cluster cluster, and the cluster contained tables, you would use the following SQL command:

-

DROP CLUSTER lot\_cluster INCLUDING TABLES;

The next lesson wraps up this module. You will briefly review the topics covered in this module. You also can take a quiz to help identify topics that you might want to review in more detail.