

Database Programming with SQL 16-2: Indexes and Synonyms Practice Activities

# Objectives

* Define an index and its use as a schema object
* Name the conditions that cause an index to be created automatically
* Create and execute a CREATE INDEX and DROP INDEX statement
* Construct and execute a function-based index
* Construct a private and public synonym

# Vocabulary

Identify the vocabulary word for each definition below.

|  |  |
| --- | --- |
| **Confirming Index** | Confirms the existence of indexes from the USER\_INDEXES data dictionary view |
| **INDEX** | Schema object that speeds up retrieval of rows |
| **CREATE PUBLIC SYNONYM** | To refer to a table by another name to simplify access |
| **Composite Index** | An index that you create on multiple columns in a table |
| **Unique index** | The Oracle Server automatically creates this index when you define a column in a table to have a PRIMARY KEY or a UNIQUE KEY constraint |
| **function-based index** | Stores the indexed values and uses the index based on a SELECT statement to retrieve the data |
| **DROP INDEX** | Removes an index |
| **Synonym** | Gives alternative names to objects |

# Try It / Solve It

1. What is an index and what is it used for?

*Definition*: **These are schema objects which make retrieval of rows from table faster.**

  - They are meant to be efficient way to find data in database. I may like to drop an index if, queries in application are not using some index or say it is not speeding up the queries or may be table is very small. An index provides direct and fast access to row in table.

·         I should create an index if the table is large and most queries are expected to retrieve less than 2 to 4 percent of the rows.

·         I should create an index if one or more columns are frequently used together in a join condition.

*Purpose*: **An index provides direct and fast access to row in table. They provide indexed path to locate data quickly, so hereby reduce necessity of heavy disk input/output operations.**

Track usage of index:

Look into what indexes employees table has:

SELECT ucm.index\_name, ucm.column\_name, ucm.column\_position, uix.uniqueness

FROM user\_indexes uix INNER JOIN user\_ind\_columns ucm ON uix.index\_name = ucm.index\_name

WHERE ucm.table\_name = 'EMPLOYEES';

Start monitoring an index:

ALTER INDEX emp\_id\_pk MONITORING USAGE;

Note down column values in V$OBJECT\_USAGE:

SELECT \* FROM v$object\_usage WHERE index\_name = 'EMP\_ID\_PK';

Run a statement which may be using the index:

SELECT \* from employees where employee\_id = 100;

Note down column values in V$OBJECT\_USAGE:

SELECT \* FROM v$object\_usage WHERE index\_name = 'EMP\_ID\_PK';

Stop monitoring an index:

ALTER INDEX emp\_id\_pk NOMONITORING USAGE;

1. What is a ROWID, and how is it used?

**Indexes use ROWID's (base 64 string representation of the row address containing block identifier, row location in the block and the database file identifier) which is the fastest way to access any particular row.**

1. When will an index be created automatically?

**For primary/unique keys: Although unique index can   be created manually, but preferred should be by using unique/primary constraint in the table. So, it   means that primary key/unique key use already existing unique index but if index is not present already, it is created while applying unique/primary key constraint.**

**Oracle also creates index automatically for LOB storage, xmltype and materialized view.**

1. Create a nonunique index (foreign key) for the DJs on Demand column (cd\_number) in the D\_TRACK\_LISTINGS table. Use the Oracle Application Developer SQL Workshop Data Browser to confirm that the index was created.

Creating index (non-unique):

**CREATE INDEX d\_tlg\_cd\_number\_fk\_i**

**on d\_track\_listings (cd\_number);**



Verify by SQL statement:

**SELECT ucm.index\_name, ucm.column\_name, ucm.column\_position, uix.uniqueness**

**FROM user\_indexes uix INNER JOIN user\_ind\_columns ucm ON uix.index\_name = ucm.index\_name**

**WHERE ucm.table\_name = 'D\_TRACK\_LISTINGS' AND column\_name = 'CD\_NUMBER';**

Verify by object browser:

Изображение выглядит как текст

Автоматически созданное описание

1. Use the join statement to display the indexes and uniqueness that exist in the data dictionary for the DJs on Demand D\_SONGS table.

**SELECT ucm.index\_name, ucm.column\_name, ucm.column\_position, uix.uniqueness**

**FROM user\_indexes uix INNER JOIN user\_ind\_columns ucm ON uix.index\_name = ucm.index\_name**

**WHERE ucm.table\_name = 'D\_SONGS';**



1. Use a SELECT statement to display the index\_name, table\_name, and uniqueness from the data dictionary USER\_INDEXES for the DJs on Demand D\_EVENTS table.

**SELECT index\_name, table\_name,uniqueness FROM user\_indexes where table\_name = 'D\_EVENTS';**

****

1. Write a query to create a synonym called dj\_tracks for the DJs on Demand d\_track\_listings table.

**CREATE SYNONYM dj\_tracks FOR d\_track\_listings;**



1. Create a function-based index for the last\_name column in DJs on Demand D\_PARTNERS table that makes it possible not to have to capitalize the table name for searches. Write a SELECT statement that would use this index.

**SELECT ucm.index\_name, ucm.column\_name, ucm.column\_position, uix.uniqueness**

**FROM user\_indexes uix INNER JOIN user\_ind\_columns ucm ON uix.index\_name = ucm.index\_name**

**WHERE ucm.table\_name = 'D\_PARTNERS';**



Create index:

**CREATE INDEX d\_ptr\_last\_name\_idx**

**ON d\_partners(LOWER(last\_name));**



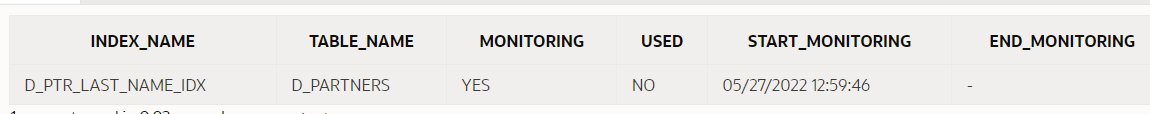
Start monitoring an index:

**ALTER INDEX d\_ptr\_last\_name\_idx MONITORING USAGE;**

****

Note down column values in V$OBJECT\_USAGE:

**SELECT \* FROM v$object\_usage WHERE index\_name = 'D\_PTR\_LAST\_NAME\_IDX';**



Run a statement which may be using the index:

**SELECT \***

**FROM d\_partners**

**WHERE LOWER(last\_name) = 'something';**



Note down column values in V$OBJECT\_USAGE:

**SELECT \* FROM v$object\_usage WHERE index\_name = 'D\_PTR\_LAST\_NAME\_IDX';**



Stop monitoring an index:

**ALTER INDEX d\_ptr\_last\_name\_idx NOMONITORING USAGE;**



1. Create a synonym for the D\_TRACK\_LISTINGS table. Confirm that it has been created by querying the data dictionary.

See problem 7 above for creation part too.

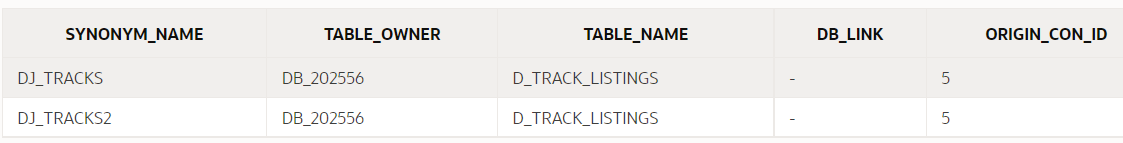
CREATE SYNONYM dj\_tracks FOR d\_track\_listings;



**CREATE SYNONYM dj\_tracks2 FOR d\_track\_listings;**



**SELECT \* FROM user\_synonyms WHERE table\_NAME = UPPER('d\_track\_listings');**

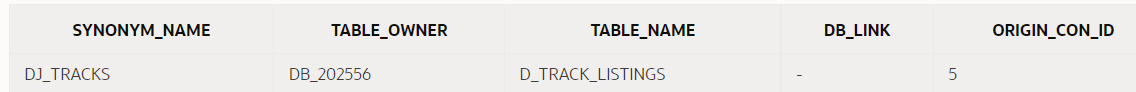


1. Drop the synonym that you created in question 9.

**DROP SYNONYM dj\_tracks2;**



SELECT \* FROM user\_synonyms WHERE table\_NAME = UPPER('d\_track\_listings');



Copyright © 2020, Oracle and/or its affiliates. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.