

```
SQL> CREATE TABLE big_dwh_table (id NUMBER, album_id NUMBER, artist_id NUMBER, country_id NUMBER, format_id NUMBER, release_date DATE, total_sales NUMBER);
```

Table created.

```
SQL> CREATE SEQUENCE dwh_seq;
```

Sequence created.

```
SQL> create or replace procedure pop_big_dwh_table as
```

```
2 v_id          number;
3 v_artist_id   number;
4 begin
5   for v_album_id in 1..10000 loop
6     v_artist_id:= ceil(dbms_random.value(0,100));
7     for v_country_id in 1..100 loop
8       select dwh_seq.nextval into v_id from dual;
9       insert into big_dwh_table values (v_id, v_album_id, v_artist_id, vcountry_id,
10        ceil(dbms_random.value(0,4)), trunc(sysdate-mod(v_id,ceil(dbms_random.value(0,1000)))),
11        ceil(dbms_random.value(0,500000)));
12     end loop;
13   end loop;
14 /
```

Procedure created.

```
SQL> exec pop_big_dwh_table
```

PL/SQL procedure successfully completed.

```
SQL> SELECT index_name, leaf_blocks, clustering_factor FROM user_indexes WHERE index_name like 'BIG_DWH%ALBUM_ID_I';
```

INDEX_NAME	LEAF_BLOCKS	CLUSTERING_FACTOR
BIG_DWH_ALBUM_ID_I	2090	4957

*** Note the index has an excellent Clustering Factor as data is insert in album_id order

```
SQL> CREATE INDEX big_dwh_album_id_i ON big_dwh_table(album_id);
```

Index created.

```
SQL> SELECT * FROM big_dwh_table WHERE album_id IS NOT NULL ORDER BY album_id;
```

1000000 rows selected.

Execution Plan

```
-----
| Id | Operation | Name | Rows | Bytes | Cost(%CPU) | Time |
-----
```

	0		SELECT STATEMENT				1000K	28M	7145	2)	00:00:48
	1		TABLE ACCESS BY INDEX ROWID	BIG_DWH_TABLE		1000K	28M	7145	2)	00:00:48	
*	2		INDEX FULL SCAN		BIG_DWH_ALBUM_ID_I		1000K	2130	2)	00:00:15	

Statistics

0 recursive calls

0 db block gets

139919 consistent gets

7040 physical reads

0 redo size

37476493 bytes sent via SQL*Net to client

733722 bytes received via SQL*Net from client

66668 SQL*Net roundtrips to/from client

0 sorts (memory)

0 sorts (disk)

1000000 rows processed