Database Programming with SQL

16-1: Working with Sequences

1. Using CREATE TABLE AS subquery syntax, create a seq_d_songs table of all the columns in the DJs on Demand database table d_songs. Use the SELECT * in the subquery to make sure that you have copied all of the columns.

```
1  SELECT * FROM seq_d_songs
```

Results | Explain | Describe | Saved SQL | History

ID	TITLE	DURATION	ARTIST	TYPE_CODE
45	Its Finally Over	5 min	The Hobbits	12
46	Im Going to Miss My Teacher	2 min	Jane Pop	12
47	Hurrah for Today	3 min	The Jubilant Trio	77
48	Meet Me At the Altar	6 min	Bobby West	1
49	Lets Celebrate	8 min	The Celebrants	77
50	All These Years	10 min	Diana Crooner	88

2. Because you are using copies of the original tables, the only constraints that were carried over were the NOT NULL constraints. Create a sequence to be used with the primary-key column of the seq_d_songs table. To avoid assigning primary-key numbers to these tables that already exist, the sequence should start at 100 and have a maximum value of 1000. Have your sequence increment by 2 and have NOCACHE and NOCYCLE. Name the sequence seq_d_songs_seq.

```
1  CREATE SEQUENCE seq_d_songs_seq
2  START WITH 100
3  INCREMENT BY 2
4  MAXVALUE 1000
5  NOCACHE
6  NOCYCLE;
```

Results | Explain | Describe | Saved SQL | History

```
Sequence created.
```

3. Query the USER_SEQUENCES data dictionary to verify the seq_d_songs_seq SEQUENCE settings.

```
1  SELECT *
2  FROM user_sequences
3  WHERE sequence_name = 'SEQ_D_SONGS_SEQ';
```

Results | Explain | Describe | Saved SQL | History

SEQUENCE_NAME	MIN_VALUE	MAX_VALUE	INCREMENT_BY	CYCLE_FLAG	ORDER_FLAG	CACHE_SIZE	LAST_NUMBER	SCALE_FLA
SEQ_D_SONGS_SEQ	1	1000	2	N	N	0	100	N

4. Insert two rows into the seq_d_songs table. Be sure to use the sequence that you created for the ID column. Add the two songs shown in the graphic.

ID	TITLE	DURATION	ARTIST	TYPE_CODE
	Island Fever	5 min	Hawaiian Islanders	22
	Castle of Dreams	4 min	The Wanderers	77

```
1  Select * from seq_d_songs
```

Results | Explain | Describe | Saved SQL | History

45	Its Finally Over	5 min	The Hobbits	12
46	Im Going to Miss My Teacher	2 min	Jane Pop	12
47	Hurrah for Today	3 min	The Jubilant Trio	77
48	Meet Me At the Altar	6 min	Bobby West	1
49	Lets Celebrate	8 min	The Celebrants	77
50	All These Years	10 min	Diana Crooner	88
102	Castle of Dreams	4 min	The Wanderers	77
100	Island Fever	5 min	Hawaiian Islanders	22

5. Write out the syntax for seq_d_songs_seq to view the current value for the sequence. Use the DUAL table. (Oracle Application Developer will not run this query.)

```
1  SELECT seq_d_songs_seq.CURRVAL
2  FROM dual;
```

Results | Explain | Describe | Saved SQL | History

ORA-08002: sequence SEQ_D_SONGS_SEQ.CURRVAL is not yet defined in this session

6. What are three benefits of using SEQUENCEs?
   - Unique Identifier
   - Automatic Value Generation
   - Efficiency

7. What are the advantages of caching sequence values?
   - Performance Improvement
   - Reduced Disk I/O
   - Faster Transactions

8. Name three reasons why gaps may occur in a sequence?
    - Rollback of Transactions
    - Manual Sequence Adjustments
    - Sequence Caching

16-2: Indexes and Synonyms

1. What is an index and what is it used for?
    - At the expense of more space and poorer performance for data update operations (INSERT, UPDATE, DELETE), an index is a database item that speeds up data retrieval operations on a table.
      It functions similarly to a book index, allowing you to locate the page number for a specific topic without having to look through the entire book.
      Because they offer fast access to rows, indexes are used to expedite the retrieval of rows from a table.
      For columns that are commonly used in JOIN conditions, ORDER BY clauses, and WHERE clauses, they are especially helpful.
2. What is a ROWID, and how is it used?
    - A database table's rows are uniquely identified by their ROWIDs.
      It indicates where the row is physically located in the database.
      Oracle uses ROWIDs internally to swiftly and effectively identify rows.
    - SELECT ROWID, first_name, last_name FROM employees WHERE ROWID = 'AAADzVAAEAAAAFkAAB';

3. When will an index be created automatically?
    - Oracle automatically generates a unique index in order to impose unique constraints or primary keys on a table.
    - When a cluster with indexed columns is created.
    - When query results are stored in a materialized view.
4. 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.

```
1  SELECT index_name, table_name
2  FROM user_indexes
3  WHERE table_name = 'D_TRACK_LISTINGS';
```

Results | Explain | Describe | Saved SQL | History

INDEX_NAME	TABLE_NAME
D_TLG_PK	D_TRACK_LISTINGS
IDX_CD_NUMBER	D_TRACK_LISTINGS
SONG_ID_IDX	D_TRACK_LISTINGS

5. Use the join statement to display the indexes and uniqueness that exist in the data dictionary for the DJs on Demand D_SONGS table.

```
1  SELECT i.index_name, i.uniqueness, ic.column_name
2  FROM user_indexes i
3  JOIN user_ind_columns ic
4  ON i.index_name = ic.index_name
5  WHERE ic.table_name = 'D_SONGS';
```

Results | Explain | Describe | Saved SQL | History

INDEX_NAME	UNIQUENESS	COLUMN_NAME
D_SNG_ID_PK	UNIQUE	ID

6. 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.

```
1  SELECT index_name, table_name, uniqueness
2  FROM user_indexes
3  WHERE table_name = 'D_EVENTS';
```

Results | Explain | Describe | Saved SQL | History

INDEX_NAME	TABLE_NAME	UNIQUENESS
D_EVE_ID_PK	D_EVENTS	UNIQUE

7. Write a query to create a synonym called dj_tracks for the DJs on Demand d_track_listings table.

```
1  CREATE SYNONYM dj_tracks FOR D_TRACK_LISTINGS;
```

Results | Explain | Describe | Saved SQL | History

Synonym created.

8. 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.
    - CREATE INDEX idx_last_name_lower ON D_PARTNERS (LOWER(last_name));

9. Create a synonym for the D_TRACK_LISTINGS table. Confirm that it has been created by querying the data dictionary.

```
1  SELECT synonym_name, table_name
2  FROM user_synonyms
3  WHERE table_name = 'D_TRACK_LISTINGS';
```

Results | Explain | Describe | Saved SQL | History

SYNONYM_NAME	TABLE_NAME
DJ_TRACKS	D_TRACK_LISTINGS
TRACK_LISTINGS	D_TRACK_LISTINGS

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

```
1  DROP SYNONYM track_listings;
```

Results | Explain | Describe | Saved SQL

Synonym dropped.