EXERCISE 13

Creating Views

1. What are three uses for a view from a DBA's perspective?

Security: Restricting access to specific columns or rows within a table.
Simplification: Simplifying complex queries by encapsulating them within a view.
Data Aggregation: Providing aggregated data, such as summaries or statistics, without exposing the raw data.

2. Create a simple view called view_d_songs that contains the ID, title and artist from the DJs on Demand table for each "New Age" type code. In the subquery, use the alias "Song Title" for the title Column.

```
CREATE VIEW view_d_songs AS
SELECT ID, title AS "Song Title", artist
FROM d_songs
WHERE type_code = 'New Age';
```

3. SELECT * FROM view_d_songs. What was returned?

```
SELECT * FROM view_d_songs;
```

4. REPLACE view_d_songs. Add type_code to the column list. Use aliases for all columns.
Or use alias after the CREATE statement as shown.

```
CREATE OR REPLACE VIEW view_d_songs AS
SELECT ID AS "Song ID", title AS "Song Title", artist AS "Artist Name",
type_code AS "Type Code"
FROM d_songs
WHERE type_code = 'New Age';
```

5. Jason Tsang, the disk jockey for DJs on Demand, needs a list of the past events and those planned for the coming months so he can make arrangements for each event's equipment setup. As the company manager, you do not want him to have access to the price that clients paid for their events. Create a view for Jason to use that displays the name of the event, the event date, and the theme description. Use aliases for each column name.

```
CREATE VIEW view_events_for_jason AS
SELECT name AS "Event Name", event_date AS "Event Date", theme_description AS "Theme"
FROM d_events;
```

6. It is company policy that only upper-level management be allowed access to individual employee salaries. The department managers, however, need to know the minimum, maximum, and average salaries, grouped by department. Use the Oracle database to prepare a view that displays the needed information for department managers.

```
CREATE VIEW view_dept_salaries AS
SELECT department_id,
    MIN(salary) AS "Min Salary",
    MAX(salary) AS "Max Salary",
    AVG(salary) AS "Avg Salary"
FROM employees
GROUP BY department_id;
```

DML Operations and Views

Use the DESCRIBE statement to verify that you have tables named copy_d_songs, copy_d_events, copy_d_cds, and copy_d_clients in your schema. If you don’t, write a query to create a copy of each.

1. Query the data dictionary USER_UPDATABLE_COLUMNS to make sure the columns in the base tables will allow UPDATE, INSERT, or DELETE. All table names in the data dictionary are stored in Uppercase.

Use the same syntax but change table_name of the other tables.

```
DESC copy_d_songs;
DESC copy_d_events;
DESC copy_d_cds;
DESC copy_d_clients;
```

2. Use the CREATE or REPLACE option to create a view of all the columns in the copy_d_songs table called view_copy_d_songs.

```
SELECT * FROM USER_UPDATABLE_COLUMNS
WHERE table_name = 'COPY_D_SONGS';
```

3. Use view_copy_d_songs to INSERT the following data into the underlying copy_d_songs table. Execute a SELECT * from copy_d_songs to verify your DML command. See the graphic.

ID TITLE DURATION ARTIST TYPE_CODE

88 Mello Jello 2 The What 4

```
CREATE OR REPLACE VIEW view_copy_d_songs AS
SELECT * FROM copy_d_songs;
```

4. Create a view based on the DJs on Demand COPY_D_CDS table. Name the view read_copy_d_cds. Select all columns to be included in the view. Add a WHERE clause to restrict the year to 2000. Add the WITH READ ONLY option.

```
INSERT INTO view_copy_d_songs (ID, TITLE, DURATION, ARTIST, TYPE_CODE)
VALUES (88, 'Mello Jello', 2, 'The What', 4);
SELECT * FROM copy_d_songs;
```

5. Using the read_copy_d_cds view, execute a DELETE FROM read_copy_d_cds WHERE cd_number= 90;

```
CREATE VIEW read_copy_d_cds AS
SELECT * FROM copy_d_cds
WHERE year = 2000
WITH READ ONLY;
```

6. Use REPLACE to modify read_copy_d_cds. Replace the READ ONLY option with WITH CHECK OPTION CONSTRAINT ck_read_copy_d_cds. Execute a SELECT * statement to verify

that the view exists.

```
DELETE FROM read_copy_d_cds WHERE cd_number = 90;
-- This will fail due to the READ ONLY option.
```

7. Use the read_copy_d_cds view to delete any CD of year 2000 from the underlying copy_d_cds.

```
CREATE OR REPLACE VIEW read_copy_d_cds AS
SELECT * FROM copy_d_cds
WHERE year = 2000
WITH CHECK OPTION CONSTRAINT ck_read_copy_d_cds;
SELECT * FROM read_copy_d_cds;
```

8. Use the read_copy_d_cds view to delete cd_number 90 from the underlying copy_d_cds table.

```
DELETE FROM read_copy_d_cds WHERE year = 2000;
```

9. Use the read_copy_d_cds view to delete year 2001 records.

```
DELETE FROM read_copy_d_cds WHERE cd_number = 90;
```

10. Execute a SELECT * statement for the base table copy_d_cds. What rows were deleted?

```
SELECT * FROM copy_d_cds;
```

11.What are the restrictions on modifying data through a view?

Cannot modify data through a view that includes GROUP BY, DISTINCT, JOIN,
or aggregate functions.Views with READ ONLY or CHECK OPTION constraints
restrict DML operations.

12. What is Moore’s Law? Do you consider that it will continue to apply indefinitely? Support your opinion with research from the internet.

Moore's Law is the observation made by Gordon Moore, co-founder of Intel, in 1965, which states that the number of transistors on a microchip doubles approximately every two years,
though the cost of computers is halved.

13. What is the “singularity” in terms of computing?

The singularity, in the context of computing and artificial intelligence, refers to a hypothetical future point where technological growth becomes uncontrollable and irreversible, resulting in unfathomable changes to human civilization.

Managing Views

1. Create a view from the copy_d_songs table called view_copy_d_songs that includes only the title and artist. Execute a SELECT * statement to verify that the view exists.

```
CREATE VIEW view_copy_d_songs AS
SELECT title, artist
FROM copy_d_songs;
SELECT * FROM view_copy_d_songs;
```

2. Issue a DROP view_copy_d_songs. Execute a SELECT * statement to verify that the view has been Deleted.

```
DROP VIEW view_copy_d_songs;
SELECT * FROM view_copy_d_songs;
-- This will return an error as the view no longer exists.
```

3. Create a query that selects the last name and salary from the Oracle database. Rank the salaries from highest to lowest for the top three employees.

```
SELECT last_name, salary
FROM (SELECT last_name, salary, RANK() OVER (ORDER BY salary DESC) AS rnk
    FROM employees)
WHERE rnk <= 3;
```

4. Construct an inline view from the Oracle database that lists the last name, salary, department ID, and maximum salary for each department. Hint: One query will need to calculate maximum salary by department ID.

```
SELECT e.last_name, e.salary, e.department_id, d.max_salary
FROM employees e
JOIN (SELECT department_id, MAX(salary) AS max_salary
    FROM employees
    GROUP BY department_id) d
ON e.department_id = d.department_id;
```

5. Create a query that will return the staff members of Global Fast Foods ranked by salary from lowest to highest.

```
SELECT last_name, salary
FROM employees
ORDER BY salary ASC;
```

Indexes and Synonyms

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

An index improves data retrieval speed by providing quick access to rows based on the indexed columns.

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

ROWID is a unique identifier for each row's physical location in the database.

3. When will an index be created automatically?

Indexes are automatically created for PRIMARY KEY and UNIQUE constraints.

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 Express SQL Workshop Data Browser to confirm that the index was created.

```
CREATE INDEX idx_cd_number ON d_track_listings (cd_number);
```

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.

```
SELECT index_name, uniqueness
FROM USER_INDEXES
WHERE table_name = 'D_SONGS';
```

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.

```
SELECT index_name, table_name, uniqueness
FROM USER_INDEXES
WHERE table_name = 'D_EVENTS';
```

7. 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;
```

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));
SELECT * FROM d_partners WHERE LOWER(last_name) = 'smith';
```

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

```
SELECT synonym_name
FROM USER_SYNONYMS
WHERE synonym_name = 'DJ_TRACKS';
```

10.Drop the synonym that you created in question

```
DROP SYNONYM dj_tracks;
```