EXERCISE 13

Creating Views

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

* Simplify complex queries for users
* Restrict access to specific columns or rows
* Present data in a logical format without changing base tables

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 djs-on-demand WHERE type_code = `New Age';

3. SELECT * FROM view_d_songs. What was returned?

It returns rows showing id, Song Title and artist for all
songs where type_code = 'New Age'.

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 (Song_ID,
Song_Title, Artist_Name, Type_code) AS SELECT id, title, artist,
type_code FROM djs_on_demand 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 event_vu AS SELECT event_name AS "Event",
  event_date AS "Date", theme_desc AS "Theme"
FROM 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 dept_salary_vu AS SELECT dept_id AS "Dept",
   MIN(salary) AS "min_salary", MAX(salary) AS "max_salary",
   AVG(salary) AS "avg-salary" FROM employees
GROUP BY dept-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.

```
SELECT table_name, column_name, insertable, updatable, deletable
FROM user_updatable_columns WHERE table_name='COPY_D_SONGS';
```

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

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.

```
CREATE OR REPLACE VIEW view-copy-d-songs AS
SELECT * FROM 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

```
INSERT INTO view-copy-d-songs VALUES (88,'Mello Jello',2,
 'The What',4);

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.

```
CREATE OR REPLACE VIEW read-copy-d-cds AS SELECT *
 FROM copy-d-cds WHERE year = 2000 WITH READ ONLY;
```

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

```
DELETE FROM read_copy-d-cds WHERE cd_number = 90;
```

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.

```
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;
```

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

```
DELETE FROM read-copy-d-cds WHERE year=2000;
```

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 cd_number =90;
```

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

```
DELETE FROM read-copy-d-cds WHERE year=2001;
```

✓

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

SELECT * FROM copy-d-cds;

*Rows with year=2000 and cd_number=2000 were deleted while all other rows remain

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

* Views with READ ONLY cannot be updated or modified
* CHECK OPTION allows changes to views which meet the view condition

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 says transistors double every 2 years in a processor.
* It is slowing down due to physical and cost limits so it wont continue indefinitely

13. What is the "singularity" in terms of computing?

* Singularity is a point at which AI or computing grows beyond human control leading to unpredictable technological change

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 OR REPLACE 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;
```

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 employees
ORDER BY salary DESC FETCH FIRST 3 ROWS ONLY;
```

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_sal FROM
employees e JOIN (SELECT department_id, MAX(salary) AS max_sal
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 name, salary FROM staff ORDER BY salary ASC;
```

Indexes and Synonyms

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

* Index is a database object that speeds up data retrieval by creating quick lookup path for rows based on column values

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

* ROWID is a unique identifier for each row in a table, It shows exact physical location on disk and helps locate rows fast

3. When will an index be created automatically?

* It will be created when PRIMARY KEY or UNIQUE is defined

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_cdnum_fk 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 i.index_name, i.uniqueness, FROM user_indexes i
JOIN user_tables t ON i.table_name = t.table_name
WHERE i.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_upper_lastname ON d_partners(UPPER(last_name));
SELECT * FROM d_partners WHERE UPPER(last_name)='SMITH';
```

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

```
CREATE SYNONYM track_list FOR d_track_listings ;
SELECT synonym_name, table_name FROM user_synonyms
WHERE synonym_name='TRACK_LIST';
```

10.Drop the synonym that you created in question

```
DROP SYNONYM track_list;
```

✓

Evaluation Procedure	Marks awarded
Query(5)	
Execution (5)	
Viva(5)	
Total (15)	
Faculty Signature	