

2/9/25

EXERCISE 13

Creating Views

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

1. Security - restrict means of specific column or row access.

2. Data consistency - provide a consistent, unchanging interface even if underlying tables or structure change.

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 New New Age songs
Select id,

title as "Song title"
from djs on demand where type_code = 'New Age'

3. SELECT * FROM view_d_songs. What was returned?

this query will return all rows from view
showing only id, song title but only for new

4. REPLACE view_d_songs. Add type_code to the column list. Use aliases for all columns.

Create or replace view view_d_songs as
select id as "Song ID",
title as "Song title",
type_code as type code;

Or use alias after the CREATE statement as shown.

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_event_list as select  
event_name as 'Event Name',  
event_date as 'event date',  
event_description as 'Theme Description'
```

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_dpt_salary as  
select department_id as 'Department ID',  
min(salary) as 'Minimum Salary',  
max(salary) as 'Maximum Salary',  
avg(salary) as 'Average 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.

```
select table_name, column_name,
       insertable, updatable, deletable
from user_updatable_columns
where table_name = 'COPY_D_SONGS'
       copy_d_events copy_d_cds, copy_d_clients;
```

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 (id, title, duration,
                                artist, type_code)
```

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 readcopy_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;

```
select from read-copy-d-cds  
as select *  
from copy_d_cds  
with check option  
constraint  
ck_readcopy_d_cds;  
select * from readcopy_d_cds;
```

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.

```
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 readcopy_d_cds  
where cd-number = 90;
```

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

```
Delete from readcopy_d_cds  
where year = 2001
```

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

```
select 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;

All rows from copy_d_cds where year=2000 were deleted;

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

You can't change or delete data through a view if the view is based on multiple tables. Views with group function or calculations can't be multiplied. Only single table based on one table can always be updated.

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

In power about every two year. It has been how for many decades but is slowing down now. It has law for many chips are approaching size limits.

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

The "singularity" is when computers become smarter than humans. At that point, AI could no longer cause very fast & major change in technology.

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;
```

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_salary  
from employee e  
join
```

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

```
select staff_name, salary  
from global_fast_foods  
order by salary asc;
```


Indexes and Synonyms

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

An index is a database object that improves the speed of data retrieval. It works like the index in a book - allowing the database to find rows faster without scanning the entire table.

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

ROWID is a unique address for each row in an Oracle table. It tells the exact physical storage location of the row on disk. Quickly occurs specific rows. Identifies duplicate rows.

3. When will an index be created automatically?

Oracle automatically creates an index when a primary key or unique constraint is created on a column.

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);  
  
SELECT index_name, table_name  
FROM user_indexes, WHERE table_name = '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.

```
SELECT i.index_name, table_name, uniqueness  
FROM user_indexes i  
JOIN user_indexes c  
ON i.index_name = c.index_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  
  
SELECT * FROM dj_tracks
```

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) = UPPER('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_listing_syn FOR
D_TRACK_LISTING;
SELECT SYNONYM_NAME, TABLE_OWNER, TABLE_NAME
FROM USER_SYNONYM WHERE SYNONYM_NAME

10. Drop the synonym that you created in question

DROP SYNONYM track_listing_syn;
SELECT SYNONYM_NAME
FROM USER_SYNONYM
WHERE SYNONYM_NAME = 'TRACK_LISTING_SYN';

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