Lab 5
Views and Roles

CSE 4308

DATABASE MANAGEMENT SYSTEMS LAB

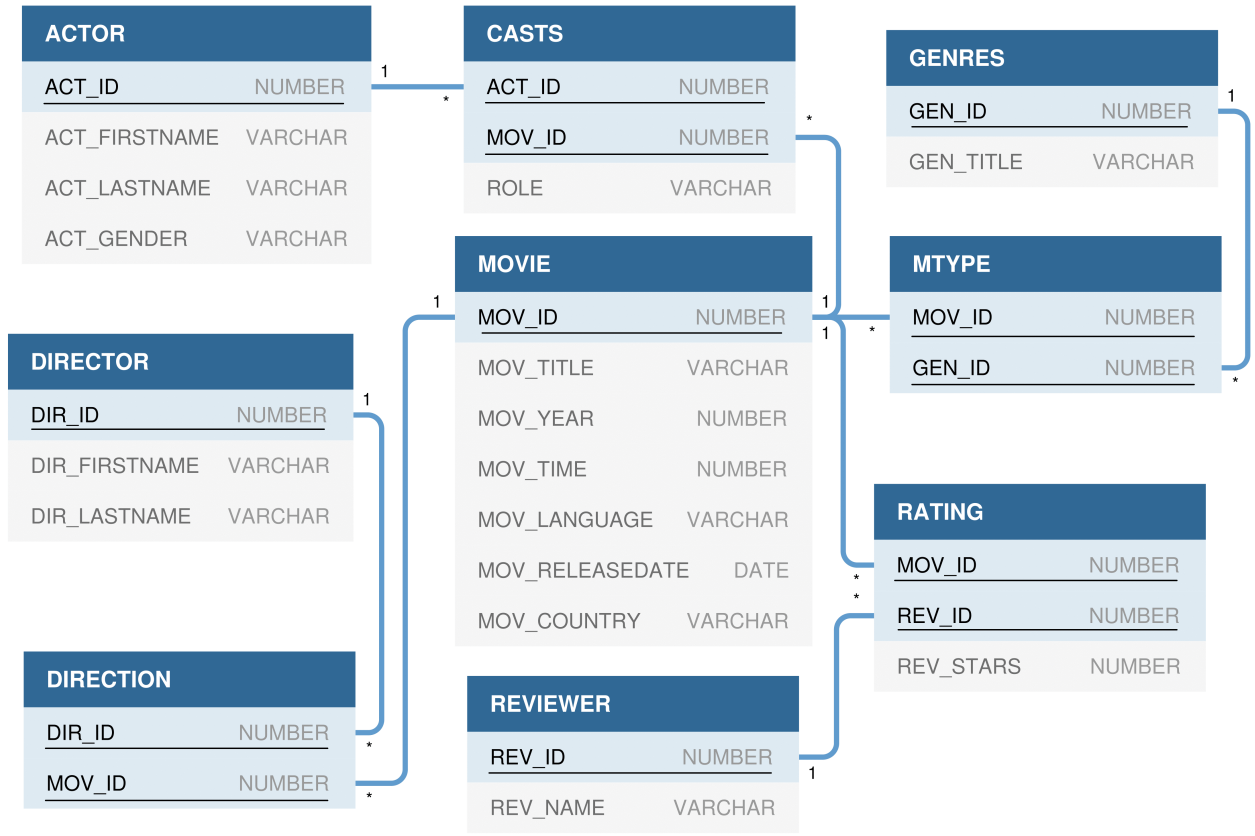
SEPTEMBER 15, 2022

1 View

An Oracle VIEW, in essence, is a virtual table that does not physically exist. Rather, it is created by a query joining one or more tables. It is stored in Oracle data dictionary. It derives its data from the tables on which it is based. Views are very powerful and handy since they can be treated just like any other table but do not occupy the space of a table. The syntax for creating view in Oracle is:

```
CREATE [OR REPLACE] VIEW view_name AS
    SELECT columns
    FROM tables
    [WHERE conditions];
```

Unlike a table, a view does not store any data. To be precise, a view only behaves like a table. And it is just a named query stored in the database. This is why sometimes a view is referred to as a named query.
Consider the following schema for a movie database:



If we want to create a view showing showing only the ID, title, and language of the movies, we can create a view as follows:

```
CREATE OR REPLACE VIEW MOVIE_SUMMARY
AS
SELECT MOV_ID, MOV_TITLE, MOV_LANGUAGE
FROM
MOVIE;
```

Now we can query data from the view:

```
SELECT DISTINCT MOV_LANGUAGE
FROM MOVIE_SUMMARY;
```

Behind the scenes, Oracle finds the stored query associated with the name MOVIE_SUMMARY and executes the following query:

```
SELECT DISTINCT MOV_LANGUAGE
FROM (SELECT MOV_ID, MOV_TITLE, MOV_LANGUAGE
    FROM MOVIE);
```

The result set returned from the view depends on the data of the underlying table. That means, modifying data from the MOVIE table will modify the results shown by querying MOVIE_SUMMARY view.
The reason we denote a view as a virtual table is that, in addition to querying them, we can even manage their contents. These modifications affect the data in the underlying table. That means, inserting or deleting rows from the view results in the insertion or deletion of rows from the underlying table. However, there are some restrictions in such modification. They are:

- If a view is defined by a query that contains SET or DISTINCT operators, a GROUP BY clause, a group function, merging of multiple tables, or a PL/SQL function, then rows cannot be inserted into, updated in, or deleted from the base tables using the view.
- If a view is defined with WITH CHECK OPTION, then a row cannot be inserted into, or updated in, the base table (using the view), if the view cannot select the row from the base table.
- If a NOT NULL column that does not have a DEFAULT clause is omitted from the view, then a row cannot be inserted into the base table using the view.

If these constraints are not violated, we can modify views. For example,

```
INSERT INTO MOVIE_SUMMARY VALUES('935', 'Emily the Criminal',
   'English');

UPDATE MOVIE_SUMMARY
SET MOV_ID = '934'
WHERE MOV_ID = '935';

DELETE FROM MOVIE_SUMMARY WHERE MOV_TITLE = 'Emily the
   Criminal';
```

Views can be dropped using the following syntax:

```
DROP VIEW view_name;
```

One of the major use cases for views are for simplifying data retrieval. For example, the following query can be used to determine the genres where actors are preferred to actresses:

```
SELECT GEN_TITLE
FROM GENRES G
WHERE (SELECT COUNT(*)
FROM ACTOR NATURAL JOIN CASTS NATURAL JOIN MTYPE NATURAL JOIN
   GENRES
WHERE ACT_GENDER = 'M' AND GEN_TITLE = G.GEN_TITLE
GROUP BY GEN_TITLE) > (SELECT COUNT(*)
FROM ACTOR NATURAL JOIN CASTS NATURAL JOIN MTYPE NATURAL JOIN
   GENRES
WHERE ACT_GENDER = 'F' AND GEN_TITLE = G.GEN_TITLE
GROUP BY GEN_TITLE);
```

However, by adding the line:

```
CREATE OR REPLACE VIEW SEXIST_GENRES AS
```

before the query, we can create a view which can then be used to simplify other queries.
Another benefit of using views can be providing additional security layer. They help us to hide certain columns and rows from the underlying tables and expose only needed data to the appropriate users. For example, a STUDENT table may store the student ID, name, department, CGPA of different students. But we might want to share only the student ID and department of the student with the instructors.

2 Roles

Role-Based Access Control (RBAC) enables us to restrict system access to authorized users based on their assigned roles. Using the RBAC model, permissions to perform specific system operations are assigned to specific roles, and system users are granted permission to perform specific operations only through their assigned roles. This simplifies system administration because users do not need to be assigned permissions directly, and instead acquire them through their assigned roles.
The general syntax for creating and deleting roles are as follows:

```
CREATE ROLE role_name;
DROP ROLE role_name;
```

Roles can be granted privileges on specific tables:

```
GRANT privilege_name ON table_name TO role_name [WITH GRANT OPTION];
```

All privileges granted to one role can be granted to another role directly:

```
GRANT role_name_1 TO role_name_2;
```

Users can be granted a role as follows:

```
GRANT role_name TO user_name;
```

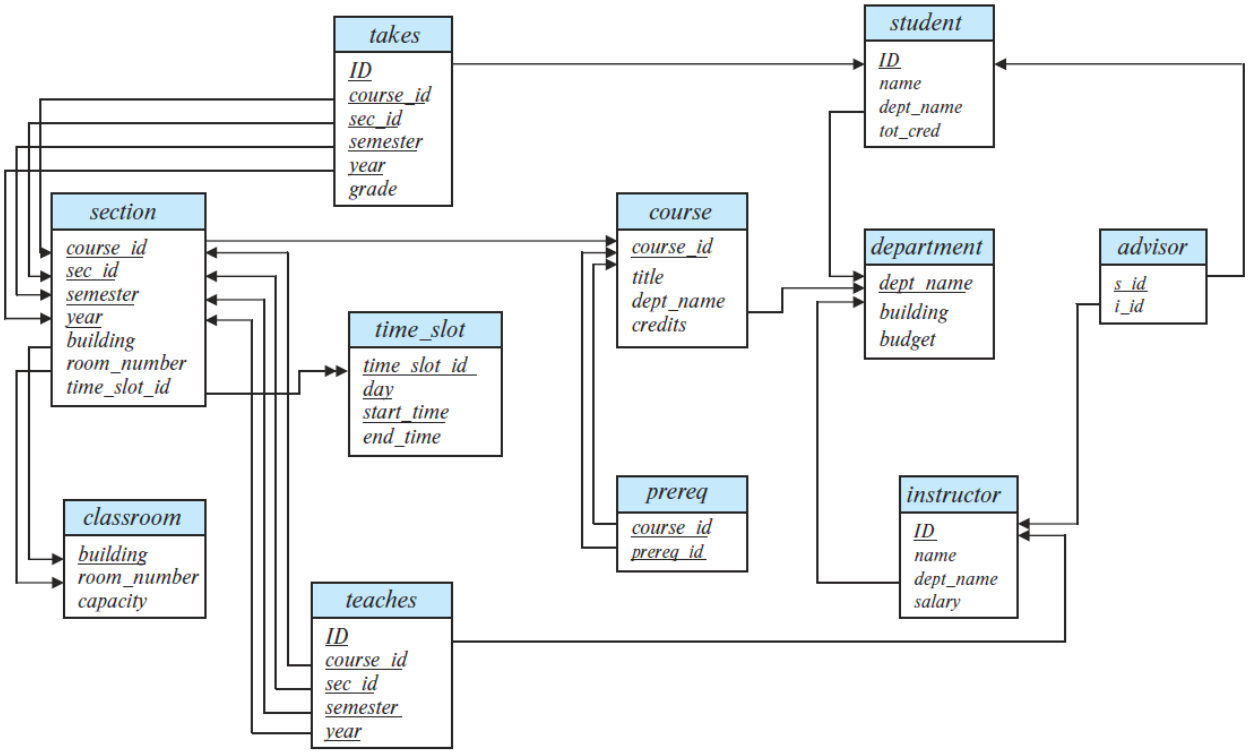
For example, we can create a role for people who sale movie tickets and grant them access to some basic information of the movies:

```
CREATE ROLE TICKET_AGENT;
GRANT SELECT ON USERNAME.MOVIE_SUMMARY TO TICKET_AGENT;

CREATE USER TA1 IDENTIFIED BY PS1;
GRANT TICKET_AGENT TO TA1;
```

3 Lab Task

Execute the `DDL+drop.sql` and `smallRelationsInsertFile.sql` scripts using command. It creates a set of tables along with values that maintain the following schema:



Here, the underlined attributes denote the primary keys and the arcs denote the foreign key relationships.
In this lab, you have to write all SQL statements in an editor first and save them with .sql extension. Then execute the SQL script.

1. Create a view named `Advisor_Selection` that shows the ID, name and department name of instructors.

2. Create another view named `Student_Count` using `Advisor_Selection` and `advisor` to show the name of the instructor and the number students assigned under them.

3. Four categories of users have been identified in the database:

   (a) Students should be able to view information regarding advisors and courses.

   (b) Course teachers should be able to view information about the students and courses.

   (c) Head of the Departments should have all the privileges that a course teacher has. Additionally, s/he should be able to add new instructors.

   (d) Administrator should be able to see information about department and instructors. They should also be able to update the department budget.

   Create roles for these categories and grant them appropriate privileges.

4. Create users under these roles and write relevant SQL queries to demonstrate that the imposed access control works.