 



Database Programming with SQL 15-3: Managing Views

Practice Activities

# Objectives

* Create and execute a query that removes a view
* Create and execute a query using an inline view
* Create and execute a top-n-analysis query

# Vocabulary

Identify the vocabulary word for each definition below.

|  |  |
| --- | --- |
| TOP-N ANALYSIS | Asks for the N largest or smallest values in a column |
| DROP | Removes a view |
| InLine View | Subquery with an alias that can be used within a SQL statement |

# Try It / Solve It

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;

1. 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

1. 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

WHERE ROWNUM <=3;

1. 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.maxsal

FROM employees e,

(SELECT department\_id, max(salary) maxsal

FROM employees

GROUP BY department\_id) d

WHERE e.department\_id = d.department\_id

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

SELECT ROWNUM, last\_name, salary

FROM

(SELECT \* FROM f\_staffs ORDER BY salary)

Extension Exercises

1. Create a new table called my\_departments and add all columns and all rows to it using a subquery from the Oracle departments table. Do a SELECT \* from my\_departments to confirm that you have all the columns and rows.

CREATE TABLE my\_departments

AS (SELECT \*

FROM departments);

SELECT \*

FROM my\_departments;

1. To view any constraints that may affect the my\_departments table, DESCRIBE my\_departments to check if any constraints were carried over from the departments table. If there are constraints on my\_departments, use an ALTER TABLE command to DISABLE all constraints on my\_departments.

DESCRIBE my\_departments;

SELECT \*

FROM user\_constraints

WHERE table\_name = 'MY\_DEPARTMENTS';

ALTER TABLE my\_departments

DISABLE CONSTRAINT SYS\_C001950111;

1. Create a view called view\_my\_departments that includes: department\_id and department\_name.

CREATE VIEW view\_my\_departments

AS (SELECT department\_id, department\_name

FROM my\_departments);

1. Add the following data to the my\_departments table using view\_my\_departments.

|  |  |
| --- | --- |
| department\_id | department\_name |
| 105 | Advertising |
| 120 | Custodial |
| 130 | Planning |

INSERT INTO view\_my\_departments

(department\_id, department\_name)

VALUES

(105, 'Advertising');

INSERT INTO view\_my\_departments

(department\_id, department\_name)

VALUES

(120, 'Custodial');

INSERT INTO view\_my\_departments

(department\_id, department\_name)

VALUES

(130, 'Planning');

1. Create or enable the department\_id column as the primary key.

ALTER TABLE my\_departments

ADD constraint department\_id\_pk PRIMARY KEY(department\_id);

1. Enter a new department named Human Resources into the my\_departments table using view\_my\_departments. Do not add a new department ID.

INSERT INTO view\_my\_departments(department\_id,department\_name)

VALUES

(0, 'Human Resources')

1. Add the Human Resources department, department ID 220, to my\_departments using view\_my\_departments.

INSERT INTO view\_my\_departments(department\_id,department\_name)

VALUES

(220, 'Human Resources')

1. Verify that the new additions to my\_departments were added using view\_my\_departments.

See chart below

SELECT\*

FROM view\_my\_departments

1. Modify view\_my\_departments to include location ID. Do a SELECT \* command to show what columns are present and a DESCRIBE command to view the columns and associated constraints.

CREATE OR REPLACE VIEW view\_my\_departments AS

SELECT department\_id, department\_name, location\_id

FROM my\_departments

DESCRIBE view\_my\_departments

1. Make location\_id a NOT NULL column in the my\_departments table.

ALTER TABLE my\_departments

MODIFY(location\_id CONSTRAINT location\_id\_nn NOT NULL) 🡪eroare fiindca a gasit valori nule, ca urmare trebuie sa modific tabelul

UPDATE my\_departments

SET location\_id = 1000

WHERE location\_id IS NULL

1. Using the Oracle database, create a complex view between locations and departments with only the following columns: department\_name, street\_address, city, and state. Include only U.S. cities. Verify that the view was created using a SELECT \* statement.

CREATE OR REPLACE VIEW complex\_view("Nume\_depart","Strada","Oras","Stat") AS

SELECT d.department\_name, l.street\_address, l.city, l.state\_province

FROM departments d JOIN locations l

USING(location\_id)

WHERE country\_id = 'US';

SELECT\*

FROM complex\_view

See chart below

|  |  |  |  |
| --- | --- | --- | --- |
| **DEPARTMENT\_ID** | **DEPARTMENT**  **\_NAME** | **MANAGER\_ID** | **LOCATION\_ID** |
| 10 | Administration | 200 | 1700 |

Results of select statement from view:

