

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** VIEW view\_name | 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;**

**SELECT \* FROM view\_copy\_d\_songs;**

**ORA-00942: table or view does not exist**

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 \* FROM**

**(SELECT last\_name, salary FROM employees ORDER BY salary  DESC)**

**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.

There may be some employees without department mentioned since it is nullable. I want to miss such records in my calculations. This is achieved in dptmx in-line view itself. Also a department without an employee is also taken in.

**SELECT empm.last\_name, empm.salary, dptmx.department\_id**

**FROM**

**(SELECT dpt.department\_id, MAX(NVL(emp.salary,0)) max\_dpt\_sal**

**FROM departments dpt LEFT OUTER JOIN employees emp ON dpt.department\_id = emp.department\_id**

**GROUP BY dpt.department\_id) dptmx LEFT OUTER JOIN employees empm ON dptmx.department\_id = empm.department\_id**

**WHERE NVL(empm.salary,0) = dptmx.max\_dpt\_sal;**

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

So, it means lowest has rank one:

**SELECT ROWNUM,last\_name, salary**

**FROM**

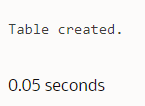
**(SELECT \* FROM f\_staffs ORDER BY SALARY);**

Extension Exercises

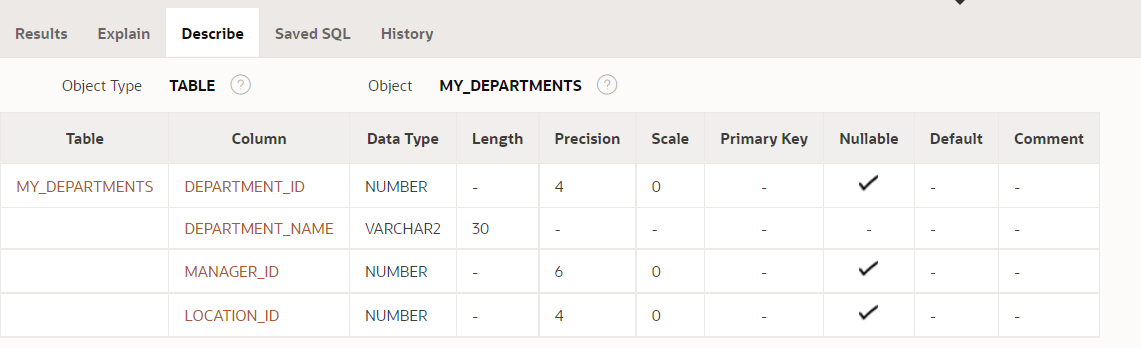
1. Create a new table called my\_departments and add all columns and all rows to it using a subquery from the Oracle departments table. Do a SELECT \* from my\_departments to confirm that you have all the columns and rows.

**CREATE TABLE my\_departments**

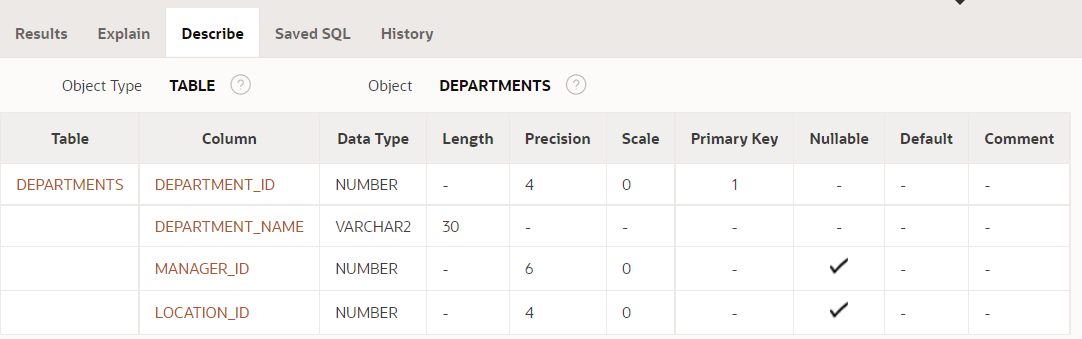
**AS ( SELECT \* FROM departments);**



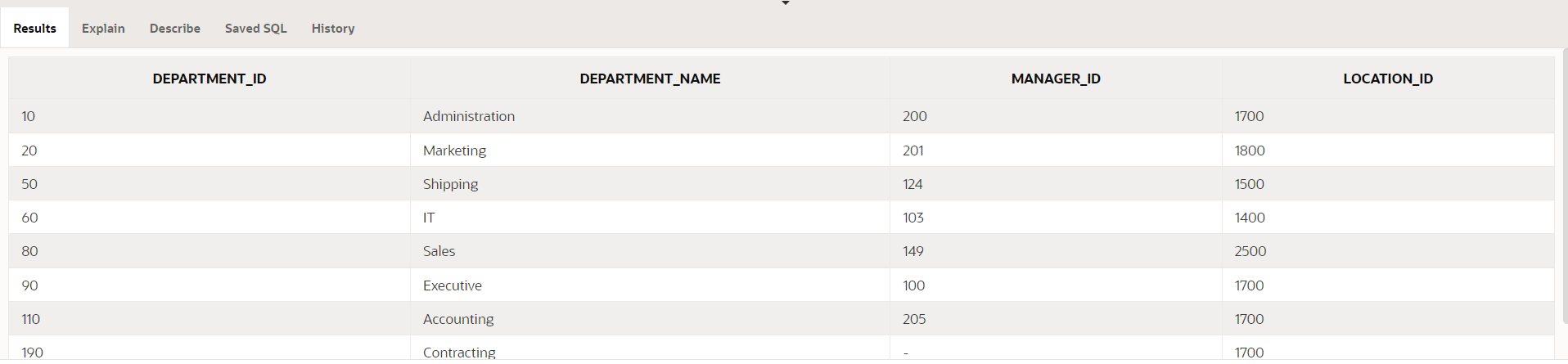
**DESCRIBE my\_departments;**



**DESCRIBE departments;**



**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.

Foreign key, check and primary key don’t go with copy but not NULL check constraint goes in copy:

**DESCRIBE my\_departments;**

I need to look into constraint\_name column in results here:

**SELECT \* FROM user\_constraints WHERE table\_name = UPPER('my\_departments');**

It suggests that I need:

**ALTER TABLE my\_departments**

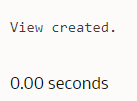
**DROP CONSTRAINT SYS\_C00868380;**

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

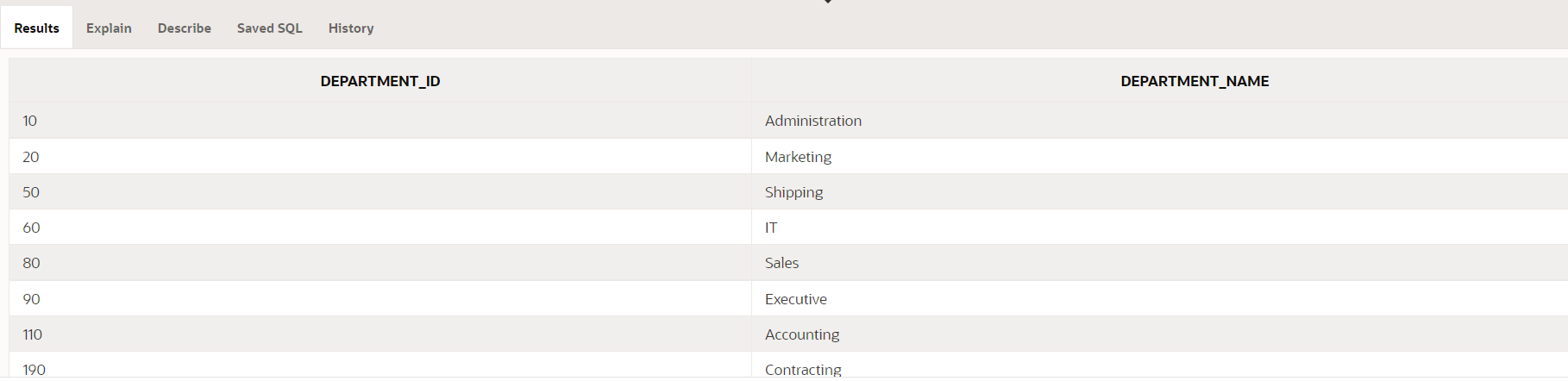
**CREATE OR REPLACE VIEW view\_my\_departments  AS**

**SELECT department\_id , department\_name**

**FROM my\_departments;**



SELECT \* FROM view\_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');**

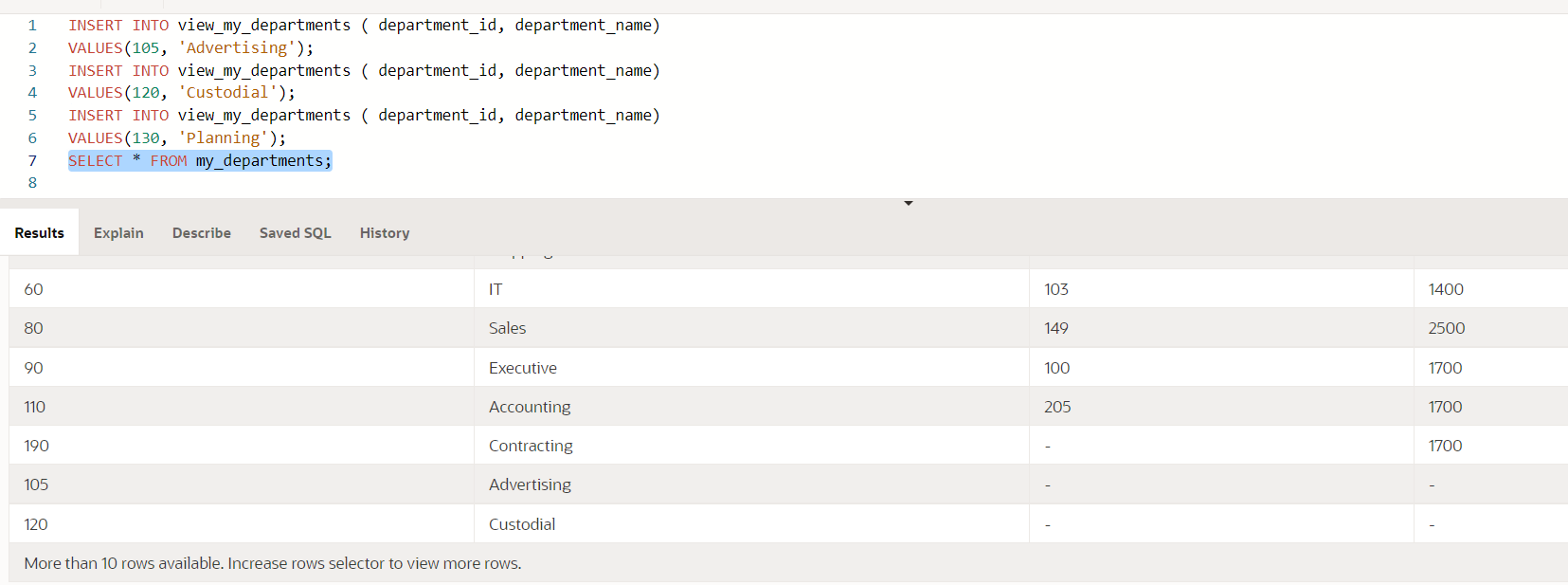
**INSERT INTO view\_my\_departments ( department\_id, department\_name)**

**VALUES(120, 'Custodial');**

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

**VALUES(130, 'Planning');**

SELECT \* FROM my\_departments;

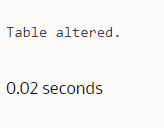


1. Create or enable the department\_id column as the primary key.

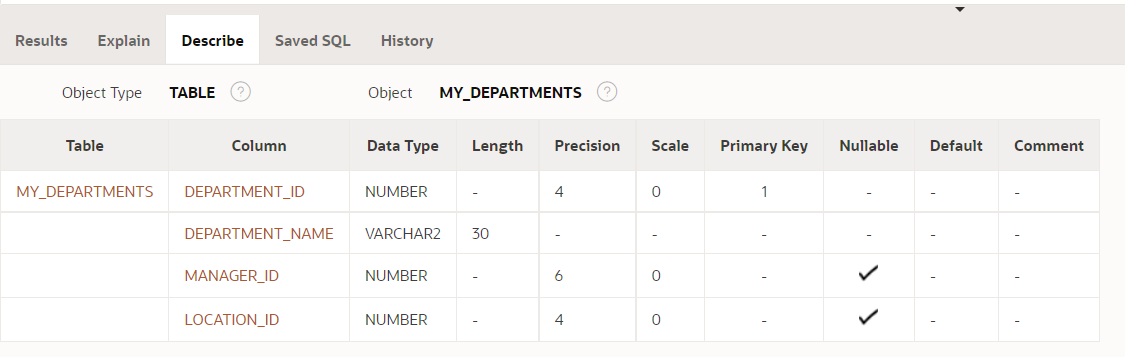
**This must be done on table itself (not the view) or I will get ORA-00942: table or view does not exist**

**ALTER TABLE my\_departments**

**ADD CONSTRAINT my\_departments\_id\_pk  PRIMARY KEY (department\_id);**



**DESCRIBE my\_departments;**



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(NULL, 'Human Resources');

**ORA-01400: cannot insert NULL into ("HKUMAR"."MY\_DEPARTMENTS"."DEPARTMENT\_ID")**

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

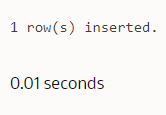
VALUES(105, 'Human Resources');

**ORA-00001: unique constraint (HKUMAR.MY\_DEPARTMENTS\_ID\_PK) violated**

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

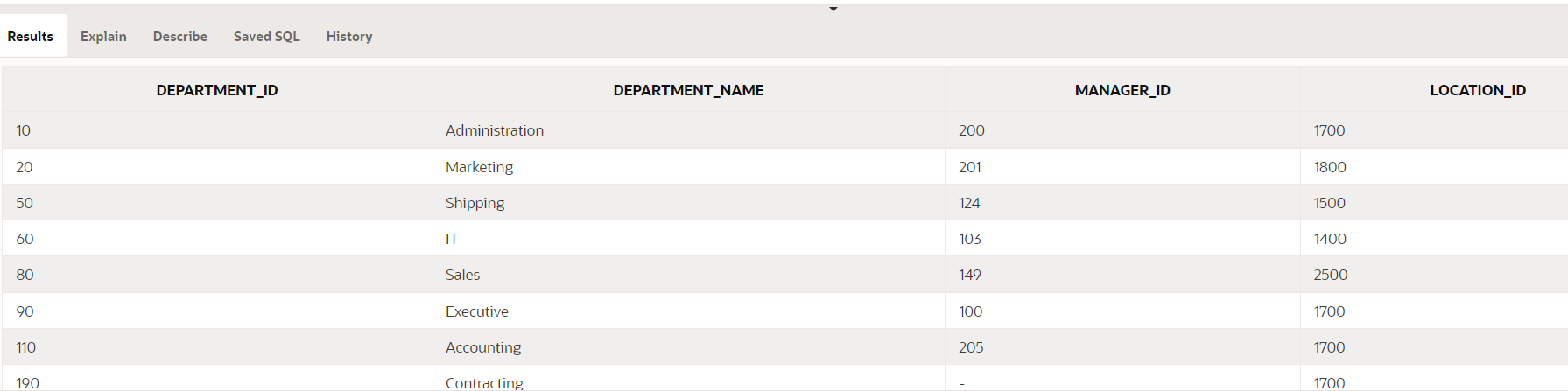
**INSERT INTO view\_my\_departments ( department\_id, department\_name)**

**VALUES(220, 'Human Resources');**



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

**SELECT \* FROM my\_departments;**

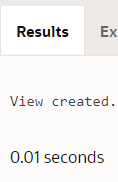


1. Modify view\_my\_departments to include location ID. Do a SELECT \* command to show what columns are present and a DESCRIBE command to view the columns and associated constraints.

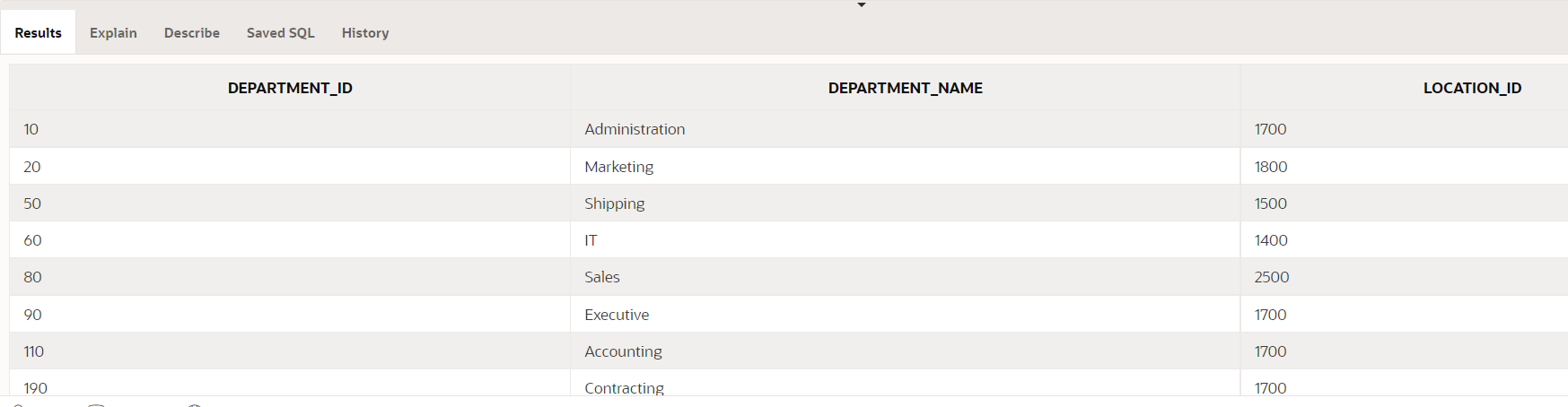
**CREATE OR REPLACE VIEW view\_my\_departments  AS**

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

**FROM my\_departments;**

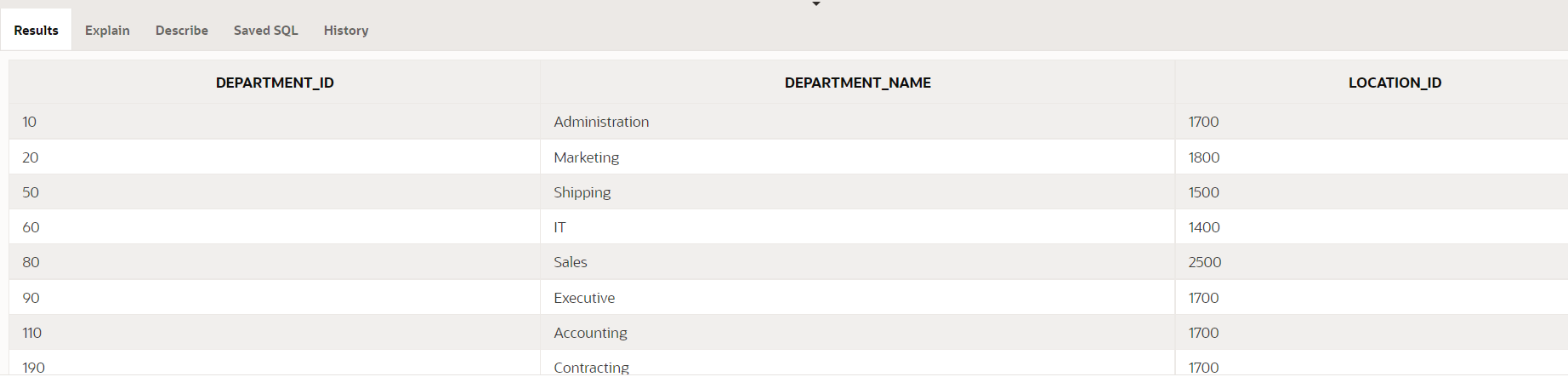


SELECT \* FROM view\_my\_departments ;

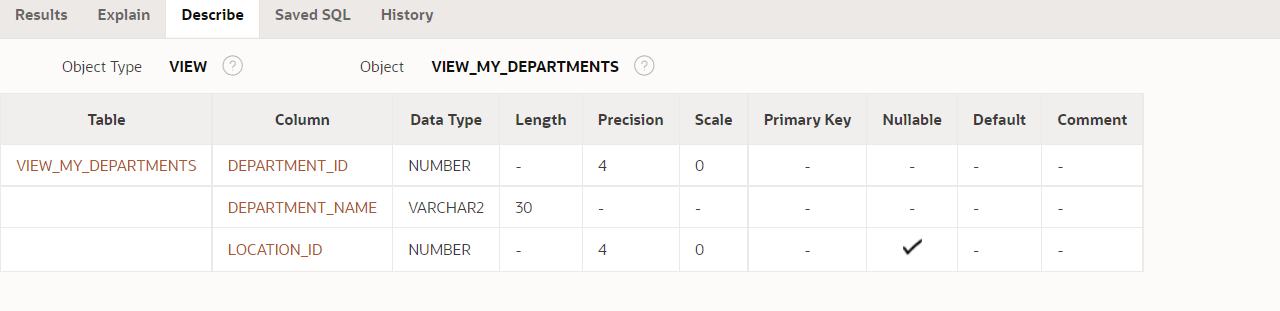


SELECT department\_id , department\_name, location\_id

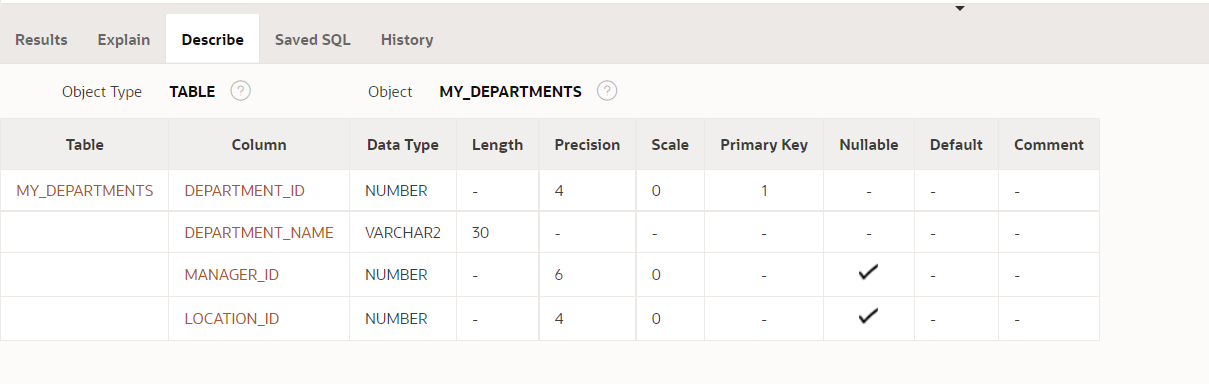
FROM my\_departments;



DESCRIBE view\_my\_departments;



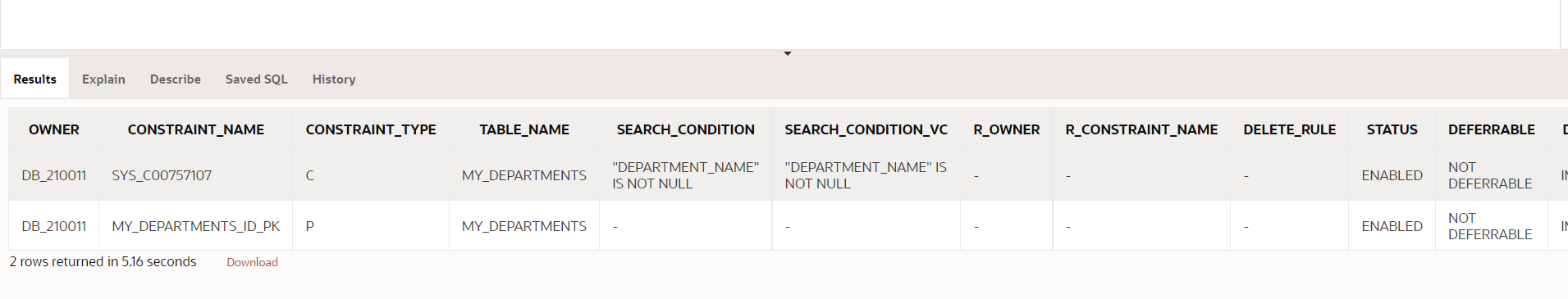
DESCRIBE my\_departments;



**So DESCRIBE take details from underlying table.**

SELECT \* FROM user\_constraints WHERE table\_name = UPPER('view\_my\_departments');

SELECT \* FROM user\_constraints WHERE table\_name = UPPER('my\_departments');



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

ALTER TABLE my\_departments

MODIFY ( location\_id NUMBER(4,0) CONSTRAINT my\_departments\_loc\_id\_nn NOT NULL); says:

**ORA-02296: cannot enable (HKUMAR.MY\_DEPARTMENTS\_LOC\_ID\_NN) - null values found**

ALTER TABLE my\_departments

ADD CONSTRAINT my\_departments\_loc\_id\_nn CHECK("LOCATION\_ID" IS NOT NULL);says:

**ORA-02293: cannot validate (HKUMAR.MY\_DEPARTMENTS\_LOC\_ID\_NN) - check constraint violated**

Now fix the error source:

UPDATE  my\_departments

SET location\_id = 1800

WHERE location\_id IS NULL;

4 row(s) updated.

I verified again above mentioned two methods of forcing NOT NULL.**Now above mentioned both ways will work. Even though practically both solve the purpose, I will prefer highlighted in green**, due to results of describe mentioned below.

Some statements used to decide highlighted is better with results:

DESCRIBE my\_departments;

SELECT \* FROM user\_constraints WHERE table\_name = UPPER('my\_departments') AND constraint\_type = 'C';

ALTER TABLE my\_departments

DROP CONSTRAINT MY\_DEPARTMENTS\_LOC\_ID\_NN;  
SELECT column\_name, nullable FROM user\_tab\_columns  
 WHERE table\_name = UPPER('my\_departments');

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.

location\_id is nullable in departments:

**CREATE OR REPLACE VIEW view\_dpt\_loc  AS**

**SELECT dpt.department\_name, loc.street\_address, loc.city, loc.state\_province**

**FROM**

**departments dpt**

**LEFT OUTER JOIN locations loc ON dpt.location\_id = loc.location\_id**

**left OUTER JOIN countries con ON loc.country\_id = con.country\_id**

**WHERE con.country\_name = 'United States of America';**

View created.

0.01 seconds

SELECT \* FROM view\_dpt\_loc;

Results of select statement from view:

