Date : 24/9/25

EXERCISE-11

AIM: To Create and understand views in SQL.

CREATING VIEWS

After the completion of this exercise, students will be able to do the following:

- Describe a view
- Create, alter the definition of, and drop a view
- Retrieve data through a view
- Insert, update, and delete data through a view
- Create and use an inline view

View

A view is a logical table based on a table or another view. A view contains no data but is like a window through which data from tables can be viewed or changed. The tables on which a view is based are called base tables.

Advantages of Views

- To restrict data access
- To make complex queries easy
- To provide data independence
- To present different views of the same data

Classification of views

1. Simple view
2. Complex view

Feature	Simple	Complex
No. of tables	One	One or more
Contains functions	No	Yes
Contains groups of data	No	Yes
DML operations thr' view	Yes	Not always

Creating a view

Syntax

CREATE OR REPLACE FORCE/NOFORCE VIEW view_name AS Subquery WITH CHECK OPTION CONSTRAINT constraint WITH READ ONLY CONSTRAINT constraint;

FORCE - Creates the view regardless of whether or not the base tables exist.

NOFORCE - Creates the view only if the ase table exist.

WITH CHECK OPTION CONSTRAINT-specifies that only rows accessible to the view can be inserted or updated.

WITH READ ONLY CONSTRAINT-ensures that no DML operations can be performed on the view.

Example: 1 (Without using Column aliases)

Create a view EMPVU80 that contains details of employees in department80.

- Group By clause
- Distinct keyword
- Columns contain by expressions
- NOT NULL columns in the base table that are not selected by the view

Example: (Using the WITH CHECK OPTION clause)

```
CREATE OR REPLACE VIEW empvu20
AS    SELECT *
FROM employees
WHERE department_id=20
WITH CHECK OPTION CONSTRAINT empvu20_ck;
```

Note: Any attempt to change the department number for any row in the view fails because it violates the WITH CHECK OPTION constraint.

Example – (Execute this and note the error)

```
UPDATE empvu20 SET department_id=10 WHERE employee_id=201;
```

Denying DML operations

Use of WITH READ ONLY option.
Any attempt to perform a DML on any row in the view results in an oracle server error.

Try this code:

```
CREATE OR REPLACE VIEW empvu10(employee_number, employee_name,job_title)
AS SELECT employee_id, last_name, job_id
FROM employees
WHERE department_id=10
WITH READ ONLY;
```

Find the Solution for the following:

1. Create a view called EMPLOYEE_VU based on the employee numbers, employee names and department numbers from the EMPLOYEES table. Change the heading for the employee name to EMPLOYEE.

```
Create OR Replace VIEW EMPLOYEES_VU (emp_id,
employee, dept_id) AS select emp_id, l_name, dept_id
FROM employees;
```

2. Display the contents of the EMPLOYEES_VU view.

```
Select * FROM employees_vu;
```

3. Select the view name and text from the USER_VIEWS data dictionary views.

Select view_name, text FROM USER_VIEWS;

4. Using your EMPLOYEES_VU view, enter a query to display all employees names and department.

Select employee, dept_id FROM EMPLOYEES_VU;

5. Create a view named DEPT50 that contains the employee number, employee last names and department numbers for all employees in department 50.Label the view columns EMPNO, EMPLOYEE and DEPTNO. Do not allow an employee to be reassigned to another department through the view.

Create view dept50(EmpNO, Employee, DEPTNO) AS Select emp_id, l_name, dept_id from employees where dept_id = 50 with CHECK OPTION;

6. Display the structure and contents of the DEPT50 view.

DESC DEPT50; Select * FROM DEPT50;

7. Attempt to reassign Matos to department 80.

Update DEPT50 SET DeptNO=80 where employee = 'Matos';

8. Create a view called SALARY_VU based on the employee last names, department names, salaries, and salary grades for all employees. Use the Employees, DEPARTMENTS and JOB_GRADE tables. Label the column Employee, Department, salary, and Grade respectively.

Create or Replace view Salary_VU (Employee, Department, Salary, Grade) AS select e.l_name, d.dept_name, e.salary, j.grade_level FROM EMPLOYEES e JOIN ~~depa~~ DEPARTMENTS d ON e.dept_id JOIN JOB_GRADES j ON e.salary BETWEEN j.lowest_sal and j.highest_sal;

Evaluation Procedure	Marks awarded
Query(5)	5
Execution (5)	5
Viva(5)	2
Total (15)	12
Faculty Signature	

RESULT:

Thus the views are studied in mysql.

Practice Problems -I

Join Clauses

Use the Oracle database for problems 1-6.

1. Join the Oracle database locations and departments table using the location_id column. Limit the results to location 1400 only.

Select l.location_id, l.city, d.dept_id, d.dept_name, FROM locations l JOIN departments d ON l.location_id = d.location_id where location_id = 1400;

2. Join DJs on Demand d_play_list_items, d_track_listings, and d_cds tables with the JOIN USING syntax. Include the song ID, CD number, title, and comments in the output.

Select pli.song_id, cds.cd_number, tl.title, tl.comments FROM d_play_list_items pli JOIN d_track_listings tl USING (track_id) JOIN d_cds cds USING (cd_id);

3. Display the city, department name, location ID, and department ID for departments 10, 20, and 30 for the city of Seattle.

Select l.city, d.dept_name, l.location_id, d.dept_id FROM departments d JOIN locations l on d.location_id = l.location_id where l.city = 'Seattle' AND d.dept_id IN (10,20,30);

4. Display country name, region ID, and region name for Americas.

Select c.country_name, r.region_id, r.region_name from countries c JOIN regions r ON c.region_id = r.region_id where r.region_name = 'Americas';

5. Write a statement joining the employees and jobs tables. Display the first and last names, hire date, job id, job title, and maximum salary. Limit the query to those employees who are in jobs that can earn more than $12,000.

Select e.first_name, e.last_name, e.hire_date, j.job_id, j.job_title, j.max_salary FROM employees e JOIN jobs j ON e.job_id = j.job_id where j.max_salary > 12000;

Inner versus Outer Joins

Use the Oracle database for problems 1-7.

1. Return the first name, last name, and department name for all employees including those employees not assigned to a department.

Select e.f_name, e.l_name, d.dept_name FROM employees e LEFT OUTER JOIN departments d on e.dept_id = d.dept_id;

2. Return the first name, last name, and department name for all employees including those departments that do not have an employee assigned to them.

Select e.f_name, e.l_name, d.dept_name FROM employees e RIGHT OUTERJOIN departments d on e.dept_id = d.dept_id;

3. Return the first name, last name, and department name for all employees including those departments that do not have an employee assigned to them and those employees not assigned to a department.

Select e.f_name, e.l_name, d.dept_name FROM employees e FULL OUTER JOIN departments d ON e.dept_id = d.dept_id;

4. Create a query of the DJs on Demand database to return the first name, last name, event date, and description of the event the client held. Include all the clients even if they have not had an event scheduled.

Select c.f_name, c.l_name, e.event_date, e.description FROM D_CLIENTS c LEFT OUTER JOIN D_EVENTS e ON c.client_id = e.client_id;

5. Using the Global Fast Foods database, show the shift description and shift assignment date even if there is no date assigned for each shift description.

Select s.shift_description, a.shift_assignment_date, FROM GFF_SHIFTS s LEFT OUTER JOIN GFF_Assignments a ON s.shift_id = a.shift_id;

Self Joins and Hierarchical Queries

For each problem, use the Oracle database.

1. Display the employee's last name and employee number along with the manager's last name and manager number. Label the columns: Employee, Emp#, Manager, and Mgr#, respectively.

Select e.l_name AS "Employee", e.emp_id AS "Emp #", m.l_name AS "Manager", m.emp_id AS "Mgr #" FROM employees e JOIN employees m ON e.manager_id = m.emp_id;

2. Modify question 1 to display all employees and their managers, even if the employee does not have a manager. Order the list alphabetically by the last name of the employee.

Select e.l_name AS "Employee", e.emp_id AS "EMP#", m.l_name AS "Manager", m.emp_id AS "Mgr#" FROM employees e LEFT OUTER JOIN employees m ON e.manager_id = e.emp_id ORDER BY e.l_name;

3. Display the names and hire dates for all employees who were hired before their managers, along with their managers' names and hire dates. Label the columns Employee, Emp Hired, Manager, and Mgr Hired, respectively.

Select e.l_name AS "Employee", e.hire_date AS "Emp Hired", m.l_name AS "Manager", m.hire_date AS "Mgr Hired" FROM employees e JOIN employees m ON e.manager_id = m.emp_id where e.hire_date < m.hire_date;

4. Write a report that shows the hierarchy for Lex De Haans department. Include last name, salary, and department id in the report.

Select l_name, salary, dept_id FROM employees START WITH l_name = 'De Haan' CONNECT BY PRIOR emp_id = manager_id;

5. What is wrong in the following statement:

SELECT last_name, department_id, salary
FROM employees
START WITH last_name = 'King'
CONNECT BY PRIOR manager_id = employee_id;

Select last_name, department_id, salary FROM employees START WITH last_name = 'King' CONNECT BY PRIOR employee_id = manager_id;

6. Create a report that shows the organization chart for the entire employee table. Write the report so that each level will indent each employee 2 spaces. Since Oracle Application Express cannot display the spaces in front of the column, use - (minus) instead.

Select LPAD('-', 2 * (Level - 1), '-') || l_name AS "Org Chart", emp_id, mgr_id FROM employees START with mgr_id is Null CONNECT PRIOR emp_id = mgr_id;

7. Re-write the report from 6 to exclude De Haan and all the people working for him.

Select LPAD('-', 2*(Level-1), '-')|| l_name AS "Org Chart", emp_id, mgr_id FROM employees where l_name != 'De Haan' Start with mgr_id is NULL CONNECT BY PRIOR emp_id = mgr_id AND prior l_name != 'De Haan';

Oracle Equijoin and Cartesian Product

1. Create a Cartesian product that displays the columns in the d_play_list_items and the d_track_listings in the DJs on Demand database.

Select * FROM d_play_list_items CROSS JOIN d_track_listings;

2. Correct the Cartesian product produced in question 1 by creating an equijoin using a common column.

Select * FROM d_play_list_items JOIN d_track_listings USING(track_id);

3. Write a query to display the title, type, description, and artist from the DJs on Demand database.

Select t.title, tt.type_description AS type, t.description, a.artist_name AS artist FROM d_tracks t JOIN d_track_types tt ON t.track_type_id = tt.track_type_id JOIN d_artists a ON t.artist_id = a.artist_id;

4. Rewrite the query in question 3 to select only those titles with an ID of 47 or 48.

Select t.title, tt.type_description AS type, t.description, a.artist_name AS artist FROM d_tracks t JOIN d_track_types tt ON t.track_type_id = tt.track_type_id JOIN d_artists a ON t.artist_id = a.artist_id where t.track_id IN (47, 48);

5. Write a query that extracts information from three tables in the DJs on Demand database, the d_clients table, the d_events table, and the d_job_assignments table.

Select c.f_name AS client_f_name, c.l_name AS client_l_name, e.event_date, ja.job_assignment_id, ja.job_description FROM d_clients c JOIN d_events e ON c.client_id = e.client_id JOIN d_job_assignments ja ON e.event_id = ja.event_id;

STDDEV - Returns the standard deviation of a column - Select STDDEV(salary) FROM employees;

SUM - Returns the sum of all values - Select SUM (salary) FROM employees;

VARIANCE - Returns the variance of a column - Select variance (salary) FROM employees);

Group Functions

1. Define and give an example of the seven group functions: AVG, COUNT, MAX, MIN, STDDEV, SUM, and VARIANCE.

AVG - Returns the average value of a numerical column - Select AVG(salary) FROM employees

COUNT - Returns the number of row satisfying criteria - Select COUNT(*) FROM employees;

MAX - Returns the maximum value in a column - Select MAX(salary) FROM employees;

MIN - Returns the minimum value in a column - Select MIN (hire-date) FROM employees;

2. Create a query that will show the average cost of the DJs on Demand events. Round to two decimal places.

select ROUND(AVG(event-cost),2) AS "Average Event Cost" FROM D-EVENTS;

3. Find the average salary for Global Fast Foods staff members whose manager ID is 19.

select AVG(salary) AS "Average Salary" FROM GFF-STAFF where mgr-id = 19;

4. Find the sum of the salaries for Global Fast Foods staff members whose IDs are 12 and 9.

Select Sum (salary) AS "Total Salary" FROM GFF-STAFF where staff-id IN (12,9);

Using the Oracle database, select the lowest salary, the most recent hire date, the last name of the person who is at the top of an alphabetical list of employees, and the last name of the person who is at the bottom of an alphabetical list of employees. Select only employees who are in departments 50 or 60

Select MIN (salary) AS "Lowest salary", MAX(hire-date) AS "Most Recent Hire Date", MIN(last-name) AS "Top last Name (A-Z)", MAX(last-name) AS "Bottom last Name (A-Z)" FROM Employees where dept-id IN(50, 60);

5. Your new Internet business has had a good year financially. You have had 1,289 orders this year. Your customer order table has a column named total_sales. If you submit the following query, how many rows will be returned?

SELECT sum(total_sales) fFROM orders;

1 rows will be returned.

6. You were asked to create a report of the average salaries for all employees in each division of the company. Some employees in your company are paid hourly instead of by salary. When you ran the report, it seemed as though the averages were not what you expected—they were much higher than you thought! What could have been the cause?

The most common cause when an average is higher than expected is that a significant group of low-paid employee were missing their salary data, and these were not counted in the denominator.

7. Employees of Global Fast Foods have birth dates of July 1, 1980, March 19, 1979, and March 30, 1969. If you select MIN(birthdate), which date will be returned?

The date that will be returned is March 30, 1969

8. Create a query that will return the average order total for all Global Fast Foods orders from January 1, 2002, to December 21, 2002.

select AVG(order-total) AS "Average order Total 2002"
FROM OFF-ORDERS WHERE order-date BETWEEN
DATE '2002-01-01' AND DATE '2002-12-21';

9. What was the hire date of the last Oracle employee hired?

Select MAX(hire-date) AS "Last Hire Date" FROM employee;

10. Your new Internet business has had a good year financially. You have had 1,289 orders this year. Your customer order table has a column named total_sales. If you submit the following query, how many rows will be returned?

SELECT sum(total_sales)
FROM orders;

This query will return 1 row.

Evaluation Procedure	Marks awarded
Practice Evaluation (5)	5
Viva(5)	2
Total (10)	7
Faculty Signature	

Practice Problems -II

COUNT, DISTINCT, NVL

1. How many songs are listed in the DJs on Demand D_SONGS table?

Select Count (*) FROM D_SONGS;

2. In how many different location types has DJs on Demand had venues?

Select COUNT(Distinct location_type) FROM D_locations;

3. The d_track_listings table in the DJs on Demand database has a song_id column and a cd_number column. How many song IDs are in the table and how many different CD numbers are in the table?

Select Count (song_id) AS "Total Song IDs" Count (Distinct cd_number) AS "Different CD Numbers" FROM d_track_listings;

4. How many of the DJs on Demand customers have email addresses?

Select COUNT(email_address) FROM D_CUSTOMERS;

5. Some of the partners in DJs on Demand do not have authorized expense amounts (auth_expense_amt). How many partners do have this privilege?

Select COUNT(auth_expense_amt) FROM D_Partners;

6. What values will be returned when the statement below is issued?

ID	type	shoe_color
456	oxford	brown
463	sandal	tan
262	heel	black
433	slipper	tan

```
SELECT COUNT(shoe_color),
COUNT(DISTINCT shoe_color)
FROM shoes;
```

Values returned: 4 (for COUNT(shoe_color)) and 3 (for COUNT(Distinct shoe_color)).

7. Create a query that will convert any null values in the auth_expense_amt column on the DJs on Demand D_PARTNERS table to 100000 and find the average of the values in this column. Round the result to two decimal places.

Select ROUND (AVG(NVL (auth_expense_amt, 100000)), 2)
FROM D_Partners;

8. Which of the following statements is/are TRUE about the following query?
SELECT AVG(NVL(selling_bonus, 0.10))
FROM bonuses;

_____a. The datatypes of the values in the NVL clause can be any datatype except date data. F date
_____b. If the selling_bonus column has a null value, 0.10 will be substituted. True (Definition of NVL)
_____c. There will be no null values in the selling_bonus column when the average is calculated. True
_____ d. This statement will cause an error. There cannot be two functions in the SELECT statement. False

9. Which of the following statements is/are TRUE about the following query?
SELECT DISTINCT colors, sizes
FROM items;

_____a. Each color will appear only once in the results set. False
_____b. Each size will appear only once in the results set. False
_____c. Unique combinations of color and size will appear only once in the results set. True
_____d. Each color and size combination will appear more than once in the results set. False

Using GROUP BY and HAVING Clauses

1. In the SQL query shown below, which of the following are true about this query?

a. Kimberly Grant would not appear in the results set. False

b. The GROUP BY clause has an error because the manager_id is not listed in the SELECT clause. False

c. Only salaries greater than 16001 will be in the result set. False

d. Names beginning with Ki will appear after names beginning with Ko. False

e. Last names such as King and Kochhar will be returned even if they don't have salaries > 16000. True

```
SELECT last_name, MAX(salary)
FROM employees
WHERE last_name LIKE 'K%' GROUP
BY manager_id, last_name HAVING
MAX(salary) >16000
ORDER BY last_name DESC ;
```

2. Each of the following SQL queries has an error. Find the error and correct it. Use Oracle Application Express to verify that your corrections produce the desired results.

a.
```
SELECT manager_id
FROM employees
WHERE AVG(salary) <16000
GROUP BY manager_id;
```

Select manager_id FROM employees GROUP BY manager_id HAVING AVG(salary) < 16000;

b.
```
SELECT cd_number, COUNT(title)
FROM d_cds
WHERE cd_number < 93;
```

select cd_number, count(title) FROM d_cds where cd_number < 93 GROUP BY cd_number;

c.
```
SELECT ID, MAX(ID), artist AS Artist FROM d_songs
WHERE duration IN('3 min', '6 min', '10 min')
HAVING ID < 50
GROUP by ID;
```

select MAX(s.ID), s.artist AS Artist FROM d_songs s where s.duration IN ('3min', '6min', '10min') GROUP BY s.artist HAVING MAX(s.ID) < 50;

d.
```
SELECT loc_type, rental_fee AS Fee
FROM d_venues
WHERE id <100
GROUP BY "Fee"
ORDER BY 2;
```

select loc_type, rental_fee AS Fee FROM d_venues where ID < 100 GROUP BY loc_type, rental_fee ORDER BY 2;

3. Rewrite the following query to accomplish the same result

SELECT DISTINCT MAX(song_id)
FROM d_track_listings WHERE
track IN (1, 2, 3);

Select MAX (song_id)
FROM d_track_listings where track IN (1,2,3);

4. Indicate True or False

a. If you include a group function and any other individual columns in a SELECT clause, then each individual column must also appear in the GROUP BY clause. True
b. You can use a column alias in the GROUP BY clause. False
c. The GROUP BY clause always includes a group function. False

5. Write a query that will return both the maximum and minimum average salary grouped by department from the employees table.

Select MAX (dept_avg_salary) AS "Max Avg Salary",
MIN (dept_avg_salary) AS "Min Avg Salary" FROM
(Select AVG(salary) AS dept_avg_salary FROM employees GROUP
BY dept_id);

6. Write a query that will return the average of the maximum salaries in each department for the employees table.

Select AVG (dept_max_salary) AS "Avg of Max Salaries"
FROM (Select MAX(salary) AS dept_max_salary FROM
employees group by dept_id;

Using Set Operators

1. Name the different Set operators?

UNION, UNIONALL, INTERSECT, MINUS

2. Write one query to return the employee_id, job_id, hire_date, and department_id of all employees and a second query listing employee_id, job_id, start_date, and department_id from the job_history table and combine the results as one single output. Make sure you suppress duplicates in the output.

select emp-id, job-id, hire-date, dept-id FROM employees UNION select emp-id, job-id, start-date, dept-id FROM job-history;

3. Amend the previous statement to not suppress duplicates and examine the output. How many extra rows did you get returned and which were they? Sort the output by employee_id to make it easier to spot.

select emp-id, job-id, hire-date, dept-id FROM employees UNION ALL select emp-id, job-id, start-date, dept-id FROM job-history order by emp-id;

There is one extra row on employee 176 with a job_id of SA_REP.

4. List all employees who have not changed jobs even once. (Such employees are not found in the job_history table)

select emp-id, l-name FROM employees where ~~employee-id~~ emp-id NOT IN(select emp-id FROM job-history);

5. List the employees that HAVE changed their jobs at least once.

select emp-id, l-name FROM employees where emp-id IN (select emp-id FROM job-history);

6. Using the UNION operator, write a query that displays the employee_id, job_id, and salary of ALL present and past employees. If a salary is not found, then just display a 0 (zero) in its place.

select emp-id, job-id, salary FROM employees UNION select emp-id, job-id, 0 FROM job-history;

Evaluation Procedure	Marks awarded
Practice Evaluation (5)	5
Viva(5)	4
Total (10)	9
Faculty Signature	

Practice Problems -III

Fundamentals of Subqueries

1. What is the purpose of using a subquery?

It is used to return data that will be used by the main query to complete it's task.

2. What is a subquery?

A subquery is a query nested inside another SQL query

3. What DJs on Demand d_play_list_items song_id's have the same event_id as song_id 45?

Select song-id FROM d-play-list-items where event_id = (select event-id FROM d-play-list-items where song-id = 45);

4. Which events in the DJs on Demand database cost more than event_id = 100?

Select event-id, cost FROM D-events where cost > (Select cost FROM D-events where event-id = 100);

5. Find the track number of the song that has the same CD number as "Party Music for All Occasions."

Select dtl.track-number FROM D-Track-Listings dtl where dtl.cd-number = (Select dc.cd-number FROM D-CDs dc where dc.title = 'Party Music for All Occasions');

6. List the DJs on Demand events whose theme code is the same as the code for "Tropical."

Select e.event-id, e.description FROM D-Events e where e.theme-code = (Select t.theme-code FROM D-Themes t where t.theme-name = 'Tropical');

7. What are the names of the Global Fast Foods staff members whose salaries are greater than the staff member whose ID is 12?

select f-name, l-name, salary FROM GFF-STAFF where salary > (Select salary FROM GFF-STAFF where staff-id = 12);

8. What are the names of the Global Fast Foods staff members whose staff types are not the same as Bob Miller's?

Select l.location_id, l.city, d.dept_name from locations l JOIN departments d on l.location_id = d.location_id where l.location_id = 1700;

9. Which Oracle employees have the same department ID as the IT department?

Select pli.song_id, cds.cd_number, dll.title, dll.comments FROM d_play_list_items pli JOIN d_track_listings dtl USING (track_id) JOIN d_cds cds USING (cd_id);

10. What are the department names of the Oracle departments that have the same location ID as Seattle?

Select l.city, d.dept_name, l.location_id, d.dept_id FROM departments d JOIN locations l ON d.location_id = l.location_id where l.city = 'Seattle' AND d.dept_id IN(10, 20, 30);

11. Which statement(s) regarding subqueries is/are true?

a. It is good programming practice to place a subquery on the right side of the comparison operator.

True

b. A subquery can reference a table that is not included in the outer query's FROM clause.

True

c. Single-row subqueries can return multiple values to the outer query.

False

Single-Row Subqueries

1. Write a query to return all those employees who have a salary greater than that of Lorentz and are in the same department as Abel.

Select * FROM employees where salary >(select salary FROM Employees where l_name = 'Lorentz') AND dept_id = (select dept_id from Employees where l_name = 'Abel');

2. Write a query to return all those employees who have the same job id as Rajs and were hired after Davies.

Select * from employees where job_id = (select job_id from employees where l_name = 'Rajs') AND hire_date > (select hire_date from employees where l_name = 'Davies');

3. What DJs on Demand events have the same theme code as event ID = 100?

select * FROM events where theme_code = (select theme_code from events where event_id = 100) AND event_id != 100;

4. What is the staff type for those Global Fast Foods jobs that have a salary less than those of any Cook staff-type jobs?

Select Distinct staff_type from JOB_INFO where staff_type = 'Global Fast Foods' AND salary < ANY (Select salary FROM JOB_INFO where staff_type = 'Cook')

5. Write a query to return a list of department id's and average salaries where the department's average salary is greater than Ernst's salary.

Select dept_id, AVG(salary) as average_department_salary from Employees GROUP BY dept_id HAVING AVG(salary) > (Select salary from employees where l_name = 'Ernst');

6. Return the department ID and minimum salary of all employees, grouped by department ID, having a minimum salary greater than the minimum salary of those employees whose department ID is not equal to 50.

select dept_id, MIN(salary) AS department_min_salary FROM employees GROUP BY dept_id HAVING MIN (salary) > (Select MIN(salary) FROM employees where dept_id != 50);

Multiple-Row Subqueries

1. What will be returned by a query if it has a subquery that returns a null?

1) In the select list (as a column): The main query proceeds as normal and the value for that column will be NULL for corresponding row.

2.) In the where clause (for comparison like =, IN, or NOT IN): The entire condition after evaluates to unknown causing the main query to return no rows.

2. Write a query that returns jazz and pop songs. Write a multi-row subquery and use the d_songs and d_types tables. Include the id, title, duration, and the artist name.

```
select m.id, m.title, m.duration, t.name AS artist_name
FROM d_movies m INNER JOIN d_types t ON
m.type_id = t.id;
```

3. Find the last names of all employees whose salaries are the same as the minimum salary for any department.

```
Select l_name FROM employees where salary IN (
Select MIN(salary) FROM employees GROUP BY dept_id);
```

4. Which Global Fast Foods employee earns the lowest salary? Hint: You can use either a single- row or a multiple-row subquery.

```
Select l_name, f_name, salary from employees
where salary = (select MIN(salary) FROM employees);
```

5. Place the correct multiple-row comparison operators in the outer query WHERE clause of each of the following:

a. Which CDs in our d_cds collection were produced before "Carpe Diem" was produced?

WHERE year <ANY (SELECT year ...

b. Which employees have salaries lower than any one of the programmers in the IT department?

WHERE salary <ANY (SELECT salary ...

c. What CD titles were produced in the same year as "Party Music for All Occasions" or "Carpe Diem"?

WHERE year IN (SELECT year ...

d. What song title has a duration longer than every type code 77 title?

WHERE duration >ALL (SELECT duration ...

6. If each WHERE clause is from the outer query, which of the following are true?

a. WHERE size > ANY -- If the inner query returns sizes ranging from 8 to 12, the value 9 could be returned in the outer query. True

b. WHERE book_number IN -- If the inner query returns books numbered 102, 105, 437, and 225 then 325 could be returned in the outer query. False

c. WHERE score <= ALL -- If the inner query returns the scores 89, 98, 65, and 72, then 82 could be returned in the outer query. False

d. WHERE color NOT IN -- If the inner query returns red, green, blue, black, and then the outer query could return white. True

e. WHERE game_date = ANY -- If the inner query returns 05-Jun-1997, 10-Dec-2002, and 2-Jan- 2004, then the outer query could return 10-Sep-2002. True

7. The goal of the following query is to display the minimum salary for each department whose minimum salary is less than the lowest salary of the employees in department 50. However, the subquery does not execute because it has five errors. Find them, correct them, and run the query.

```
SELECT department_id , MIN(salary)
FROM  employees WHERE
MIN(salary) HAVING
MIN(salary) > GROUP BY
department_id SELECT
MIN(salary) from employees
WHERE department_id < 50; => department_id = 50, department_id = 50,
```

select dept_id, MIN(salary)
from employees GROUP BY dept_id
HAVING MIN(salary) < select
MIN(salary) From employees where

No data was returned from this query.

8. Which statements are true about the subquery below?

```
SELECT employee_id, last_name
FROM employees
WHERE salary =
(SELECT MIN(salary)
FROM employees
GROUP BY department_id);
```

a. The inner query could be eliminated simply by changing the WHERE clause to WHERE MIN(salary). False

b. The query wants the names of employees who make the same salary as the smallest salary in any department. False

c. The query first selects the employee ID and last name, and then compares that to the salaries in every department. False

8. This query will not execute. True

9. Write a pair-wise subquery listing the last_name, first_name, department_id, and manager_id for all employees that have the same department_ id and manager_id as employee 141. Exclude employee 141 from the result set.

Select l_name, f_name, dept_id, manager_id FROM employees where (dept_id, manager_id = (select dept_id, manager_id from employees where emp_id = 141) AND emp_id <> 141;

10. Write a non-pair-wise subquery listing the last_name, first_name, department_id, and manager_id for all employees that have the same department_ id and manager_id as employee 141.

Select l_name, f_name, dept_id, manager_id FROM employees where dept_id = (select dept_id from employees where emp_id = 141) AND manager_id = (select manager_id from employees where emp_id = 141) AND emp_id <> 141;

Correlated Subqueries

1. Explain the main difference between correlated and non-correlated subqueries?

A correlated subquery depends upon the outer query while Non-correlated subquery is independent of the outer query.

2. Write a query that lists the highest earners for each department. Include the last_name, department_id, and the salary for each employee.

Select e.l_name, e.dept_id, e.salary from employees e where e.salary = (select MAX(e2.salary) from employees e2 where e2.dept_id = e.dept_id);

3. Examine the following select statement and finish it so that it will return the last_name, department_id, and salary of employees who have at least one person reporting to them. So we are effectively looking for managers only. In the partially written SELECT statement, the WHERE clause will work as it is. It is simply testing for the existence of a row in the subquery.

SELECT (enter columns here)
FROM (enter table name here) outer

WHERE 'x' IN (SELECT 'x'
FROM (enter table name here) inner
WHERE inner(enter column name here) = inner(enter column name here)
Finish off the statement by sorting the rows on the department_id column.

Select outer last_name, outer department_id, outer salary FROM employees outer where Exists (select 'x' FROM employees where inner. manager_id = outer employee_id) ORDER BY outer department_id;

4. Using a WITH clause, write a SELECT statement to list the job_title of those jobs whose maximum salary is more than half the maximum salary of the entire company. Name your subquery MAX_CALC_SAL. Name the columns in the result JOB_TITLE and JOB_TOTAL, and sort the result on JOB_TOTAL in descending order.

Hint: Examine the jobs table. You will need to join JOBS and EMPLOYEES to display the job_title.

With max-company-salary AS (select MAX(salary) AS MAX-TOTAL-SAL FROM employees), max-job-salary AS (select job_id, MAX(salary) AS MAX-CALC-SAL from employees GROUP BY job-id) select j. job_title, mjs. MAX-CALC-SAL AS JOB_TOTAL FROM max-job-salary mjs JOIN jobs j ON mjs. job-id = j. job-id CROSS JOIN max-company-salary mcs where mjs. MAX-CALC-SAL > (mcs. MAX-TOTAL-SAL/2) ORDER BY

JOB_TOTAL DESC;

Summarizing Queries for practice

INSERT Statements

Students should execute DESC tablename before doing INSERT to view the data types for each column. VARCHAR2 data-type entries need single quotation marks in the VALUES statement.

1. Give two examples of why it is important to be able to alter the data in a database.

To Reflect Real-world changes (updates) and to log New Information (Inserts).

2. DJs on Demand just purchased four new CDs. Use an explicit INSERT statement to add each CD to the copy_d_cds table. After completing the entries, execute a SELECT * statement to verify your work.

CD_NUMBER	TITLE	PRODUCER	YEAR
97	Celebrate the Day	R&B Inc.	2003
98	Holiday Tunes for All Ages	Tunes are Us	2004
99	Party Music	Old Town Records	2004
100	Best of Rock and Roll	Old Town Records	2004

INSERT INTO copy_d_cds (CD_Number, Title, Producer, YEAR) VALUES (97, 'Celebrate the Day', 'R&B Inc.', 2003);
INSERT IN
Select * FROM copy_d_cds where CD_Number BETWEEN 97 AND 100;

3. DJs on Demand has two new events coming up. One event is a fall football party and the other event is a sixties theme party. The DJs on Demand clients requested the songs shown in the table for their events. Add these songs to the copy_d_songs table using an implicit INSERT statement.

INSERT INTO copy_d_songs Values ('Swaying Summer', NULL, 12 /*, . . . other columns */);

INSERT INTO copy_d_songs VALUES ('Victory Victory', '5 min', 12 /*, . . . other columns */);

ID	TITLE	DURATION	TYPE_CODE
52	Surfing Summer	Not known	12
53	Victory Victory	5 min	12

4. Add the two new clients to the copy_d_clients table. Use either an implicit or an explicit INSERT.

CLIENT_NUMBER	FIRST_NAME	LAST_NAME	PHONE	EMAIL
6655	Ayako	Dahish	3608859030	dahisha@harbor.net
6689	Nick	Neuville	9048953049	nnicky@charter.net

Insert INTO copy_d_clients (client_number, first_name, last_name, Phone, Email) VALUES (6655, 'Ayako', 'Dahish', '3608859030', dahisha@harbor.net');

Insert into copy_d_clients (client_number, first_name, last_name, Phone, Email) VALUES (6689, 'Nick', 'Neuville', '9048953049', 'nicky@charter.net');

5. Add the new client's events to the copy_d_events table. The cost of each event has not been determined at this date.

ID	NAME	EVENT_DATE	DESCRIPTION	COST	VENUE_ID	PACKAGE_CODE	THEME_CODE	CLIENT_NUMBER
110	Ayako Anniversary	07-Jul-2004	Party for 50, sixties dress, decorations		245	79	240	6655
115	Neuville Sports Banquet	09-Sep-2004	Barbecue at residence, college alumni, 100 people		315	87	340	6689

~~Update copy_f_food_v~~

Insert into copy_d_events (ID, NAME, EVENT_DATE, DESCRIPTION, COST, VENUE_ID, PACKAGE_CODE, Theme_code, CLIENT_NUMBER) VALUES (110, 'Ayako Anniversary', '07-Jul-2004', 'Party for 50, Sixties dress, decorations', NULL, 245, 79, 240, 6655);

Insert into copy_d_events (ID, NAME, EVENT_DATE, Description, COST, VENUE_ID, PACKAGE_code, Theme_code, Client_Number) VALUES (115, 'Neuville Sports Banquet', '09-Sep-2004', 'Barbecue at residence, college alumni, 100 people', NULL, 315, 87, 340, 6689);

6. Create a table called rep_email using the following statement:

CREATE TABLE rep_email (id NUMBER(3) CONSTRAINT rel_id_pk PRIMARY KEY, first_name VARCHAR2(10), last_name VARCHAR2(10), email_address VARCHAR2(10))

Populate this table by running a query on the employees table that includes only those employees who are REP's.

Insert into rep_email (id, f_name, l_name, email_address) Select emp_id, ~~[illegible]~~ f_name, l_name, email_address FROM employees where job_title = 'REP';

Updating Column Values and Deleting Rows

NOTE: Copy tables in this section do not yet exist; students must create them.

If any change is not possible, give an explanation as to why it is not possible.

1. Monique Tuttle, the manager of Global Fast Foods, sent a memo requesting an immediate change in prices. The price for a strawberry shake will be raised from $3.59 to $3.75, and the price for fries will increase to $1.20. Make these changes to the copy_f_food_items table.

Update copy_f_food_items SET Price = 3.75
where item_name = 'Strawberry Shake';
Update copy_f_food_items SET Price = 1.20
where item_name = Fries;

2. Bob Miller and Sue Doe have been outstanding employees at Global Fast Foods. Management has decided to reward them by increasing their overtime pay. Bob Miller will receive an additional $0.75 per hour and Sue Doe will receive an additional $0.85 per hour. Update the copy_f_staffs table to show these new values. (Note: Bob Miller currently doesn't get overtime pay. What function do you need to use to convert a null value to 0?)

Update copy_f_staffs SET overtime_pay =
overtime_pay + 0.75 where f_name = 'Bob' AND
l_name = 'Miller';
Update copy_f_staffs SET overtime_pay
= overtime_pay + 0.85 where f_name = 'Sue'
AND l_name = 'Doe';

3. Add the orders shown to the Global Fast Foods copy_f_orders table:

ORDER_NUMBER	ORDER_DATE	ORDER_TOTAL	CUST_ID	STAFF_ID
5680	June 12, 2004	159.78	145	9
5691	09-23-2004	145.98	225	12
5701	July 4, 2004	229.31	230	12

Insert INTO copy_f_orders (ORDER_NUMBER, ORDER_DATE, ORDER_TOTAL, CUST_ID, STAFF_ID) VALUES (5680, 2004-06-01', 197.55, 145, 7);
Insert into copy_f_orders (ORDER_NUMBER, ORDER_DATE, ORDER_TOTAL, CUST_ID, STAFF_ID) VALUES (5643, '2004-04-23', 145.98, 225, 12);
Insert into copy_f_orders (Order_Number, Order_Date, Order_Total, Cust_ID, STAFF_ID) VALUES (5701, '2004-07-04', 229.31, 230, 12);

4. Add the new customers shown below to the copy_f_customers table. You may already have added Katie Hernandez. Will you be able to add all these records successfully?

ID	FIRST_NAME	LAST_NAME	ADDRESS	CITY	STATE	ZIP	PHONE_NUMBER
145	Katie	Hernandez	92 Chico Way	Los Angeles	CA	98008	8586667641
225	Daniel	Spode	1923 Silverado	Denver	CO	80219	7193343523
230	Adam	Zurn	5 Admiral Way	Seattle	WA		4258879009

Insert into copy_f_customers (ID, f_name, l_name, Address, City, State, ZIP, Phone_Number) VALUES (145, 'Katie', 'Hernandez', '92 Chico Way', 'Los Angeles', 'CA', '98008', '8586667641');
Insert into copy_f_customers (ID, f_name, l_name, Address, City, State, ZIP, Phone_Number) VALUES (225, 'Daniel', 'Spode', '1923 Silverado', 'Denver', 'CO', '80219', '7193343523');
Insert into copy_f_customers (ID, f_name, l_name, Address, City, State, ZIP, Phone_Number) VALUES (230, 'Adam', 'Zurn', '5 Admiral Way', 'Seattle', 'WA', '425887900', '4258879009');

5. Sue Doe has been an outstanding Global Foods staff member and has been given a salary raise. She will now be paid the same as Bob Miller. Update her record in copy_f_staffs.

Update copy_f_staffs set salary = (Select salary FROM copy_f_staffs where f_name = 'Bob' AND l_name = 'miller') where f_name = 'Sue' AND l_name = 'Doe';

6. Global Fast Foods is expanding their staff. The manager, Monique Tuttle, has hired Kai Kim. Not all information is available at this time, but add the information shown at right.

ID	FIRST_NAME	LAST_NAME	BIRTHDATE	SALARY	STAFF TYPE
25	Kai	Kim	3-Nov-1988	6.75	Order Taker

Insert into copy_f_staffs(ID, f_name, l_name, Birthdate, Salary, STAFF_TYPE) VALUES(25, 'Kai', 'Kim', '1988-11-03', 6.75, 'Order Taker');

7. Now that all the information is available for Kai Kim, update his Global Fast Foods record to include the following: Kai will have the same manager as Sue Doe. He does not qualify for overtime. Leave the values for training, manager budget, and manager target as null.

Update copy_f_staffs set manager_id = (Select id FROM copy_f_staffs where f_name = 'Sue' AND l_name = 'Doe'), overtime_pay = 0, training = NULL, manager_budget = NULL, manager_target = NULL;

8. Execute the following SQL statement. Record your results.

DELETE from departments
WHERE department_id = 60;

The statement will delete all rows from the departments table where the value in the department_id column is 60.

9. Kim Kai has decided to go back to college and does not have the time to work and go to school. Delete him from the Global Fast Foods staff. Verify that the change was made.

Delete FROM copy_f_staffs where id = 25;
Select * FROM copy_f_staffs where id = 25;

10. Create a copy of the employees table and call it lesson7_emp;

Once this table exists, write a correlated delete statement that will delete any employees from the lesson7_employees table that also exist in the job_history table.

Create TABLE lesson7_emp AS select * FROM employees;
Delete from lesson7_emp where exists (Select 1 from job_history jh where jh.emp_id = e.id);

DEFAULT Values, MERGE, and Multi-Table Inserts

1. When would you want a DEFAULT value?

You would want a default value in a column when you want to ensure that a column always has a value, even if a user or application doesn't explicitly provide one during an INSERT operation.

2. Currently, the Global Foods F_PROMOTIONAL_MENUS table START_DATE column does not have SYSDATE set as DEFAULT. Your manager has decided she would like to be able to set the starting date of promotions to the current day for some entries. This will require three steps:

a. In your schema, Make a copy of the Global Foods F_PROMOTIONAL_MENUS table using the following SQL statement:

Create Table copy-f-promotional-menus AS
Select * FROM Global-Foods-F-PROMOTIONAL-MENUS
Where 1=2;

b. Alter the current START_DATE column attributes using:

ALTER TABLE copy-f-promotional-menus MODIFY
START-DATE DEFAULT SYSDATE;

c. INSERT the new information and check to verify the results.

INSERT a new row into the copy_f_promotional_menus table for the manager's new promotion. The promotion code is 120. The name of the promotion is 'New Customer.' Enter DEFAULT for the start date and '01-Jun-2005' for the ending date. The giveaway is a 10% discount coupon. What was the correct syntax used?

Insert into ~~manager-copy-d-cds~~ copy-f-promotional-menus (~~cd title, cd artist~~ Promotion-code Promotion-name end-date, ~~cd year~~ giveaway) VALUES (~~'Hello world I Am', 'Middle Earth Records', 1948~~ 120, 'New customer', '01-JUN-2005', 'DD-MON-YY', '10% discount coupon');
Select * FROM copy-f-promotional-menus where promotion-code = 120;

3. Allison Plumb, the event planning manager for DJs on Demand, has just given you the following list of CDs she acquired from a company going out of business. She wants a new updated list of CDs in inventory in an hour, but she doesn't want the original D_CDS table changed. Prepare an updated inventory list just for her.

a. Assign new cd_numbers to each new CD acquired.

Insert into manager-copy-d-cds (cd-number, cd-title, cd-artist, cd-year) VALUES (CD-NUMBER-SEQ.NEXTVAL, 'New CD Title Example', 'New Artist Example', 2025);

b. Create a copy of the D_CDS table called manager_copy_d_cds. What was the correct syntax used?

Create TABLE manager-copy-d-cds AS select * FROM D_CDS where 1=2;

c. INSERT into the manager_copy_d_cds table each new CD title using an INSERT statement. Make up one example or use this data:

Insert INTO manager_copy_d_cds (cd_title, cd_artist, cd_year) VALUES ('Hello world I Am', 'middle Earth records', 1998);

20, 'Hello World Here I Am', 'Middle Earth Records', '1998' What was the correct syntax used?

d. Use a merge statement to add to the manager_copy_d_cds table, the CDs from the original table. If there is a match, update the title and year. If not, insert the data from the original table. What was the correct syntax used?

Merge INTO manager_copy_d_cds target USING D_CDS source ON (target.cd_id = source.cd_id) when matched THEN UPDATE SET target.cd_title = source.cd_title, target.cd_year = source.cd_year when NOT MATCHED THEN INSERT (cd_id, cd_title, cd_artist, cd_year) VALUES (source.cd_id, source.cd_title, source.cd_artist, source.cd_year);

4. Run the following 3 statements to create 3 new tables for use in a Multi-table insert statement. All 3 tables should be empty on creation, hence the WHERE 1=2 condition in the WHERE clause.

CREATE TABLE sal_history (employee_id, hire_date, salary) AS
SELECT employee_id, hire_date, salary
FROM employees
WHERE 1=2;
CREATE TABLE mgr_history (employee_id, manager_id, salary)
AS SELECT employee_id, manager_id, salary
FROM employees
WHERE 1=2;
CREATE TABLE special_sal (employee_id, salary) AS
SELECT employee_id, salary
FROM employees
WHERE 1=2;

Once the tables exist in your account, write a Multi-Table insert statement to first select the employee_id, hire_date, salary, and manager_id of all employees. If the salary is more than 20000 insert the employee_id and salary into the special_sal table. Insert the details of employee_id, hire_date, and salary into the sal_history table. Insert the employee_id, manager_id, and salary into the mgr_history table.

You should get a message back saying 39 rows were inserted. Verify you get this message and verify you have the following number of rows in each table:
Sal_history: 19 rows
Mgr_history: 19 rows
Special_sal: 1

Insert All when salary > 20000 THEN INTO special_sal (emp_id, salary) VALUES (emp_id, salary) VALUES (emp_id, salary) when manager_id is NOT NULL THEN INTO mgr_history (emp_id, manager_id, salary) VALUES (emp_id, manager_id, salary) ELSE INTO sal_history (emp_id, hire_date, salary) values (emp_id, hire_date, salary) select emp_id, hire_date, salary, manager_id FROM employees;

Creating Tables

1. Complete the GRADUATE CANDIDATE table instance chart. Credits is a foreign-key column referencing the requirements table.

Create Table grad_candidates (candidate_id NUMBER(6) PRIMARY KEY, f_name VARCHAR2 (50) NOT NULL, l_name VARCHAR2 (50) NOT NULL, major VARCHAR2 (50), credits_id NUMBER(3), NUMBER(3,2), graduation_date DATE, CONSTRAINT fk_credits_req FOREIGN KEY (credits_id) REFERENCES requirements (credits_id));

2. Write the syntax to create the grad_candidates table.

~~DESC grad_candidates~~

Create Table grad_candidates (candidate_id NUMBER(6) PRIMARY KEY, f_name VARCHAR2(50) NOTNULL, l_name VARCHAR2(50) NOTNULL, major VARCHAR2 (50), credits_id NUMBER(3), NUMBER (3,2), graduation_date DATE, CONSTRAINT fk_credits_req FOREIGN KEY (credits_id) REFERENCES requirements (credits_id));

3. Confirm creation of the table using DESCRIBE.

~~Create Table MYLASTNAME_table AS select *~~ ~~FROM~~ DESC grad_candidates;

4. Create a new table using a subquery. Name the new table your last name – e.g., smith_table. Using a subquery, copy grad_candidates into smith_table.

Create TABLE MYLASTNAME_table AS select * FROM grad_candidates;

5. Insert your personal data into the table created in question 4.

INSERT INTO MYLASTNAME_table (candidate_id, f_name, l_name, major, credits_id, gpa, graduation_date) VALUES (101, 'Your First Name', 'Your Last Name', 'Computer Science', 3.85, '01-MAY-2026'); select * FROM MYLASTNAME_TABLE where l_name = 'Your Last Name';

6. Query the data dictionary for each of the following:
- USER_TABLES
- USER_OBJECTS
- USER_CATALOG or USER_CAT

In separate sentences, summarize what each query will return.

Select table_name, tablespace_name FROM USER_TABLES where table_name IN ('GRAD_CANDIDATES', 'MYLASTNAME_TABLE');

Summary: This query returns a list of all tables owned by the current user in the database schema.

select obj_name, obj_type, created FROM USER_OBJECTS
object_name IN ('GRAD_CANDIDATES', 'MYLASTNAME_TABLE');

Summary: This query returns information about all objects (tables, views, indexes, sequences, etc.) owned by the current user.

select table_name, table_type FROM USER_CATALOG;

Summary: This query returns information about all tables and views owned by the current user.

Modifying a Table

Before beginning the practice exercises, execute a DESCRIBE for each of the following tables: o_employees and o_jobs. These tables will be used in the exercises. You will need to know which columns do not allow null values.

NOTE: If students have not already created the o_employees, o_departments, and o_jobs tables they should create them using the four steps outlined in the practice.

1. Create the three o_tables – jobs, employees, and departments – using the syntax:

   Create TABLE o_jobs (job_id NUMBER(4) PRIMARY KEY, job_title VARCHAR2(35) NOTNULL); Create Table o_departments (dept_id NUMBER(4), PRIMARYKEY, dept_name VARCHAR2(30), NOT NULL);

   Create TABLE o_employees (emp_id NUMBER(6) PRIMARYKEY, f_name VARCHAR2(20), l_name VARCHAR2(25) NOTNULL, email VARCHAR2(25) NOT NULL UNIQUE, hire_date DATE, salary NUMBER(8,2), job_id NUMBER(4) REFERENCES o_jobs (job_id), dept_id NUMBER(4) REFERENCES o_departments (dept_id);

2. Add the Human Resources job to the jobs table:

   Insert into o_jobs (job_id, job_title) VALUES (101, 'Human Resources');

3. Add the three new employees to the employees table:

   Insert into o_employees (emp_id, f_name, l_name, email, hire_date, job_id, dept_id) VALUES (200, 'Alex', 'smith', 'asmith', SYSDATE, 101, 103);

   3. Insert into o_employees (emp_id, f_name, l_name, email, hire_date, job_id, dept_id) VALUES (201, 'Maria', 'Garcia', 'mgarcia', SYSDATE, 101, 102);

   Insert into o_employees (emp_id, f_name, l_name, email, hire_date, job_id, dept_id) VALUES (202, 'Lee', 'Wong', 'Lwong', SYSDATE, 101, 10);

4. Add Human Resources to the departments table:

   Insert into o_departments (dept_id, dept_name,

5. Why is it important to be able to modify a table?

1. CREATE a table called Artists.

a. Add the following to the table:

- artist ID
- first name
- last name
- band name
- email
- hourly rate
- song ID from d_songs table

b. INSERT one artist from the d_songs table.

c. INSERT one artist of your own choosing; leave song_id blank.

d. Give an example how each of the following may be used on the table that you have created:

1) ALTER TABLE
2) DROP TABLE
3) RENAME TABLE
4) TRUNCATE
5) COMMENT ON TABLE

a. Explain to students how you want the DJs on Demand artist's table assignment to be completed. Students should be able to list the term followed by the SQL statement they used. For example:

2. In your o_employees table, enter a new column called "Termination." The datatype for the new column should be VARCHAR2. Set the DEFAULT for this column as SYSDATE to appear as character data in the format: February 20th, 2003.

3. Create a new column in the o_employees table called start_date. Use the TIMESTAMP WITH LOCAL TIME ZONE as the datatype.

ALTER TABLE O_employees ADD (start_date TIMESTAMP WITH LOCAL TIME ZONE);

4. Truncate the o_jobs table. Then do a SELECT * statement. Are the columns still there? Is the data still there?

Yes, The Truncate Table command only removes the rows (data) from the table. NO, the data will not be there.

5. What is the distinction between TRUNCATE, DELETE, and DROP for tables?

Truncate Removes all the rows from a table. Delete removes specific rows or all rows from a table. Drop removes the entire table.

6. List the changes that can and cannot be made to a column.

1. Change the data type
2. Change the size / precision
3. Rename the column
4. Change the default value.
5. Change the Null contraint
6. Add / Remove constraints

Changes that cannot be made

1. Decrease Datatype Size / Precision
2. Change a Datatype compatibility
3. Change NOT NULL

7. Add the following comment to the o_jobs table:
"New job description added"
View the data dictionary to view your comments.

COMMENT ON TABLE O_jobs IS 'New job description added';
Select * from user_col_comments where table_name = 'O_JOBS';

8. Rename the o_jobs table to o_job_description.

ALTER TABLE O_jobs RENAME TO O_job_description;

9.F_staffs table exercises:

A. Create a copy of the f_staffs table called copy_f_staffs and use this copy table for the remaining labs in this lesson.

Create TABLE copy_f_staffs AS
Select * FROM f_staffs;

B. Describe the new table to make sure it exists.

DESC copy_f_staffs;

C. Drop the table.

DROP TABLE copy_f_staffs;

D. Try to select from the table.

Select * FROM copy_f_staffs;

E. Investigate your recyclebin to see where the table went.

~~Select original~~ Select original_name, object_name, droptime FROM user_recyclebin where original_name = 'COPY_F_STAFFS';

a. Try to select from the dropped table by using the value stored in the OBJECT_NAME column. You will need to copy and paste the name as it is exactly, and enclose the new name in " " (double quotes). So if the dropped name returned to you is BIN$Q+x1nJdcUnngQESYELVIdQ==$0, you need to write a query that refers to "BIN$Q+x1nJdcUnngQESYELVIdQ==$0".

Select * FROM "BIN$Q+x1nJdcUnngQESYELVIdQ==$0";

b. Undrop the table.

FLASHBACK TABLE copy_f_staffs TO BEFORE DROP;

c. Describe the table.

DESC copy_f_staffs;

11. Still working with the copy_f_staffs table, perform an update on the table.

a. Issue a select statement to see all rows and all columns from the copy_f_staffs table;

Select * FROM copy_f_staffs;

b. Change the salary for Sue Doe to 12 and commit the change.

Update copy_f_staffs SET salary =12
where staff_name = 'Sue Doe';
COMMIT;

c. Issue a select statement to see all rows and all columns from the copy_f_staffs table;

Select * FROM copy_f_staffs;

d. For Sue Doe, update the salary to 2 and commit the change.

Update copy_f_staffs set salary = 2 where staff_name = 'Sue Doe'; COMMIT;

e. Issue a select statement to see all rows and all columns from the copy_f_staffs table;

Select * FROM copy_f_staffs;

f. Now, issue a FLASHBACK QUERY statement against the copy_f_staffs table, so you can see all the changes made.

Select * FROM copy_f_staffs AS OF TIMESTAMP (SYSTIMESTAMP - INTERVAL '10' MINUTE) where staff_name = 'Sue Doe';

g. Investigate the result of f), and find the original salary and update the copy_f_staffs table salary column for Sue Doe back to her original salary.

'Update copy_f_staff SET salary = 1000 where staff_name = 'Sue Doe'; COMMIT;

Evaluation Procedure	Marks awarded
Practice Evaluation (5)	5
Viva(5)	4
Total (10)	9
Faculty Signature	