DBT Practice Quiz III

Total points 20/20

The respondent's email (**anishrane1555@gmail.com**) was recorded on submission of this form.

Full Name *

Anish rane

12 Digit PRN *

250240320015

Centre * Dropdown

Kharghar

✓ SELECT name FROM table1 * 1/1
UNION ALL
SELECT name FROM table2
ORDER BY name DESC;

What does this query achieve?

- ○ Combines names from both tables, removes duplicates, and sorts descending
- ◉ Combines names from both tables, retains duplicates, and sorts descending ✓
- ○ Combines and sorts ascending without duplicates
- ○ Only combines tables without sorting

✓ SELECT name FROM employee *1/1
WHERE salary = (SELECT MAX(salary) FROM employee WHERE salary < (SELECT MAX(salary) FROM employee));

This query returns employees with:

- ○ Highest salary
- ◉ Second highest salary ✓
- ○ Lowest salary
- ○ Average salary

✓ SELECT a.name, b.name FROM employee a * 1/1
INNER JOIN employee b ON a.manager_id = b.emp_id;

This query retrieves:

- (●) Employee names alongside their manager names ✓
- ( ) Employees without managers
- ( ) Employees managing other employees
- ( ) Employees with salaries matching their managers

✓ GRANT SELECT, INSERT ON employees TO user1 WITH GRANT OPTION; * 1/1

User1 can now:

- (●) Select and Insert records and give others these permissions ✓
- ( ) Only read records from employees table
- ( ) Only insert records into employees table
- ( ) Create new tables related to employees

✓ SELECT emp_name, salary FROM employee e1 *1/1
WHERE salary > (SELECT AVG(salary) FROM employee e2 WHERE e2.dept_id = e1.dept_id);

This query returns employees who:

- ○ Earn above overall average salary
- ◉ Earn above the average salary within their own department ✓
- ○ Earn below department average salary
- ○ Are in a department with higher-than-average overall salary

✓ BEGIN TRANSACTION; * 1/1
UPDATE accounts SET balance = balance - 500 WHERE id = 1;
SAVEPOINT SP1;
UPDATE accounts SET balance = balance + 500 WHERE id = 2;
ROLLBACK TO SP1;
COMMIT;

What is the final outcome of this transaction?

- ○ Both account balances remain unchanged
- ○ Both accounts are updated permanently
- ◉ Account 1 updated; Account 2 rollbacked ✓
- ○ Account 2 updated; Account 1 rollbacked

✓ UPDATE employees SET salary = salary * 1.10 WHERE emp_id IN (SELECT emp_id FROM performance WHERE rating = 'A'); *1/1

This query does what?

- ○ Increases all employee salaries by 10%
- ◉ Gives a 10% raise to employees with rating 'A' ✓
- ○ Updates salaries to exactly 1.10 for top-rated employees
- ○ Increases salaries for employees without rating ‘A’

✓ ALTER TABLE employees ADD CONSTRAINT chk_salary CHECK (salary > 1000); *1/1

This command:

- ○ Adds a new column salary to employees
- ◉ Ensures all future employee salaries exceed 1000 ✓
- ○ Deletes employees with salary less than 1000
- ○ Updates salaries below 1000 automatically

✓ CREATE TABLE Orders (*1/1
OrderID int PRIMARY KEY,
CustomerID int,
FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID) ON DELETE CASCADE
);

If a Customer is deleted from Customers, what happens to related Orders?

- ○ Orders remain unaffected
- ○ Orders are updated with NULL CustomerID
- ◉ Orders are deleted automatically ✓
- ○ Constraint violation error occurs

✓ BEGIN TRANSACTION; * 1/1
DELETE FROM accounts WHERE balance < 0;
ROLLBACK;

What does this set of commands accomplish?

- ○ Deletes accounts with negative balances permanently
- ◉ Temporarily deletes, but then undoes deletion of negative balance accounts ✓
- ○ Commits changes immediately after deletion
- ○ Locks table accounts for further transactions permanently

✓ SELECT dept_id, COUNT(emp_id) FROM employees * 1/1
GROUP BY dept_id
HAVING COUNT(emp_id) > 5;

This query fetches:

- ○ Departments with exactly 5 employees
- ◉ Departments with more than 5 employees ✓
- ○ Employees grouped by department having less than 5 employees
- ○ Departments and their total employees

✓ SELECT department, MAX(avg_salary) *1/1
FROM (SELECT department, AVG(salary) AS avg_salary FROM employees
GROUP BY department) AS dept_avg;

What does this query return?

- ○ Average salary of each department
- ◉ Department having the highest average salary ✓
- ○ Highest salary in each department
- ○ Average salary for the entire organization

✓ SELECT e.emp_name, m.emp_name AS manager_name, d.dept_name * 1/1
FROM employee e
JOIN employee m ON e.manager_id = m.emp_id
JOIN department d ON e.dept_id = d.dept_id;

What does this query display?

- (●) Employees, their managers, and departments ✓
- ( ) Employees with no manager
- ( ) Managers only
- ( ) Departments with no employees

✓ SELECT name FROM authors a WHERE EXISTS *1/1
(SELECT * FROM books b WHERE b.author_id = a.author_id AND b.year < 2000);

Which authors are retrieved?

- ( ) Authors who published no books
- (●) Authors with books published before 2000 ✓
- ( ) Authors with all books after 2000
- ( ) All authors irrespective of publication

✓ SELECT name FROM products WHERE price > ALL (SELECT price FROM products WHERE category = 'Toys'); *1/1

This query fetches products:

- ○ With price higher than the cheapest toy
- ◉ With price higher than every toy ✓
- ○ Priced equal to toys
- ○ Cheaper than all toys

✓ SELECT d.dept_name, COUNT(e.emp_id) FROM departments d * 1/1
LEFT JOIN employees e ON d.dept_id = e.dept_id
GROUP BY d.dept_name;

Departments with zero employees will:

- ○ Be excluded
- ○ Show NULL as count
- ◉ Show count as zero ✓
- ○ Show an error

✓ UPDATE employees e SET salary = salary * 1.05 WHERE EXISTS (SELECT * FROM performance p WHERE p.emp_id = e.emp_id AND p.rating='B'); *1/1

Who receives salary updates?

- ○ Employees without performance rating
- ◉ All employees rated B ✓
- ○ Employees not rated B
- ○ All employees irrespective of rating

✓ SELECT SUM(CASE WHEN dept = 'HR' THEN salary ELSE 0 END) AS hr_salary FROM employees; *1/1

This query calculates:

- ○ Total salary of all departments
- ◉ Total salary for HR department only ✓
- ○ Salaries excluding HR department
- ○ Average HR salary

✓ SELECT emp_name, salary, RANK() OVER (ORDER BY salary DESC) as rank FROM employees;
Employees with identical salaries will have: *1/1

- ○ Different ranks, consecutive numbers
- ○ Same rank, no gaps in numbering
- ◉ Same rank, gaps in numbering ✓
- ○ No ranks assigned

✓ SELECT dept, role, COUNT(*) FROM employees GROUP BY ROLLUP(dept, role);

This query provides: *1/1

- ○ Counts grouped by dept and role separately
- ◉ Counts grouped by dept and role with subtotals ✓
- ○ Only total employees in each dept
- ○ Only grand total of employees