IT Skill Test GIC Myanmar

Duration: 30 Minutes *Total Questions: 8*

---

1. You are working on a customer management system for an e-commerce platform. The database has a 'customers' table with columns: customer_id, first_name, last_name, email, and registration_date. You need to find all customers who registered in the last 30 days and have made at least one purchase. The 'orders' table has columns: order_id, customer_id, order_date, and total_amount. Which SQL query would you use?

A. SELECT c.* FROM customers c JOIN orders o ON c.customer_id = o.customer_id WHERE c.registration_date >= SYSDATE - 30

B. SELECT DISTINCT c.* FROM customers c JOIN orders o ON c.customer_id = o.customer_id WHERE c.registration_date >= SYSDATE - 30

C. SELECT c.* FROM customers c WHERE c.registration_date >= SYSDATE - 30 AND EXISTS (SELECT 1 FROM orders o WHERE o.customer_id = c.customer_id)

D. SELECT c.* FROM customers c, orders o WHERE c.customer_id = o.customer_id AND c.registration_date >= SYSDATE - 30

2. Which SQL statement would you use to create an index on the 'email' column of the 'users' table to improve query performance?

A. CREATE INDEX idx_email ON users (email);

B. ALTER TABLE users ADD INDEX idx_email (email);

C. CREATE UNIQUE INDEX idx_email ON users (email);

D. INDEX idx_email ON users (email);

3. You are designing a database for a new e-commerce platform. The system needs to store information about products, customers, and orders. Which of the following SQL statements would be most appropriate to create a table for storing customer information?

A. CREATE TABLE customers (id INT, name VARCHAR(50), email VARCHAR(100), PRIMARY KEY (name))

B. CREATE TABLE customers (id INT PRIMARY KEY, name VARCHAR(50), email VARCHAR(100) UNIQUE)

C. CREATE TABLE customers (id INT, name VARCHAR(50), email VARCHAR(100))

D. CREATE TABLE customers (id INT AUTO_INCREMENT, name VARCHAR(50), email VARCHAR(100), PRIMARY KEY (id, email))

---

4. In a database for a library system, you need to find all books that have never been borrowed. The schema includes a 'books' table (book_id, title, author) and a 'loans' table (loan_id, book_id, borrower_id, loan_date, return_date). Which SQL query would accomplish this?

A. SELECT b.* FROM books b LEFT JOIN loans l ON b.book_id = l.book_id WHERE l.book_id IS NULL;

B. SELECT b.* FROM books b WHERE b.book_id NOT IN (SELECT book_id FROM loans);

C. SELECT b.* FROM books b WHERE NOT EXISTS (SELECT 1 FROM loans l WHERE l.book_id = b.book_id);

D. SELECT b.* FROM books b MINUS SELECT b.* FROM books b JOIN loans l ON b.book_id = l.book_id;

5. You need to write a query that returns the top 5 customers who have spent the most money in the last year. The 'orders' table has columns: order_id, customer_id, order_date, and total_amount. Which SQL query would you use?

A. SELECT customer_id, SUM(total_amount) AS total_spent FROM orders WHERE order_date >= ADD_MONTHS(SYSDATE, -12) GROUP BY customer_id ORDER BY total_spent DESC FETCH FIRST 5 ROWS ONLY;

B. SELECT customer_id, SUM(total_amount) AS total_spent FROM orders WHERE order_date >= ADD_MONTHS(SYSDATE, -12) GROUP BY customer_id ORDER BY total_spent DESC LIMIT 5;

C. SELECT customer_id, SUM(total_amount) AS total_spent FROM orders WHERE order_date >= ADD_MONTHS(SYSDATE, -12) GROUP BY customer_id HAVING ROWNUM <= 5 ORDER BY total_spent DESC;

D. SELECT customer_id, SUM(total_amount) AS total_spent FROM orders WHERE order_date >= ADD_MONTHS(SYSDATE, -12) GROUP BY customer_id ORDER BY total_spent DESC WHERE ROWNUM <= 5;

6. You are working on a web application that uses an Oracle database. The application is experiencing slow performance, and you suspect it's due to inefficient queries. Which of the following SQL statements would you use to identify the most resource-intensive queries?

A. SELECT * FROM v$sql ORDER BY cpu_time DESC

B. SELECT * FROM dba_hist_sqlstat ORDER BY elapsed_time_total DESC

C. SELECT * FROM v$sqlarea ORDER BY executions DESC

D. SELECT * FROM dba_tables ORDER BY num_rows DESC

7. You need to create a view that shows the total sales for each product category in the last month. The database has tables: 'products' (product_id, name, category_id), 'categories' (category_id, name), and 'sales' (sale_id, product_id, sale_date, amount). Which SQL statement would you use?

A. CREATE VIEW category_sales AS SELECT c.name, SUM(s.amount) AS total_sales FROM categories c JOIN products p ON c.category_id = p.category_id JOIN sales s ON p.product_id = s.product_id WHERE s.sale_date >= ADD_MONTHS(SYSDATE, -1) GROUP BY c.name;

B. CREATE VIEW category_sales AS SELECT c.name, SUM(s.amount) AS total_sales FROM categories c, products p, sales s WHERE c.category_id = p.category_id AND p.product_id = s.product_id AND s.sale_date >= ADD_MONTHS(SYSDATE, -1) GROUP BY c.name;

C. CREATE OR REPLACE VIEW category_sales AS SELECT c.name, SUM(s.amount) AS total_sales FROM categories c JOIN products p ON c.category_id = p.category_id JOIN sales s ON p.product_id = s.product_id WHERE s.sale_date >= ADD_MONTHS(SYSDATE, -1) GROUP BY c.name;

D. CREATE VIEW category_sales (category, sales) AS SELECT c.name, SUM(s.amount) FROM categories c JOIN products p ON c.category_id = p.category_id JOIN sales s ON p.product_id = s.product_id WHERE s.sale_date >= ADD_MONTHS(SYSDATE, -1) GROUP BY c.name;

8. You are optimizing a frequently used query in your web application. The current query is: SELECT * FROM orders WHERE status = 'completed' AND order_date BETWEEN '2023-01-01' AND '2023-12-31' ORDER BY total_amount DESC; Which of the following indexes would be most beneficial for this query?

A. CREATE INDEX idx_orders ON orders(status, order_date, total_amount)

B. CREATE INDEX idx_orders ON orders(total_amount, status, order_date)

C. CREATE INDEX idx_orders ON orders(status, total_amount)

D. CREATE INDEX idx_orders ON orders(order_date, status)