Comprehensive SQL and Relational Algebra Study Questions

Q1. In the relational model, a tuple corresponds to:

A) A column in a table

B) A row in a table

C) A schema

D) A foreign key

Q2. Which of the following may be be a candidate key for the student relation?

- Schema:
- student(ID CHAR(6), name VARCHAR(20), surname VARCHAR(20), cgpa NUMERIC(3,2))

A) (name, surname)

B) (cgpa)

C) (ID)

D) (name, cgpa)

Q3. Which SQL keyword removes duplicates in query results?

A) all

B) unique

C) distinct

D) remove

Q4. In relational algebra, the selection operation is denoted by:

A) π

B) σ

C) ρ

D) ×

Q5. Which of the following can be said about the SQL query below intended to “Find names of all instructors in the 'Comp. Sci.' department.”

- Schema:
- instructor(ID CHAR(5), name VARCHAR(20), dept_name VARCHAR(20), salary NUMERIC(8,2))

```
SELECT name
FROM instructor
WHERE dept_name = 'Comp. Sci.';
```

The query is:

A) Correct

B) Incorrect (missing FROM)

C) Incorrect (missing WHERE)

D) Incorrect (uses GROUP BY)

Q6. Choose the best answer:

The PRIMARY KEY constraint ensures:

A) Values are unique

B) Values are unique and not null

C) Values may be duplicated

D) Foreign key integrity only

Q7. Given the schema definitions below:

- Schema:
- instructor(ID, name, dept_name)
- teaches(ID, course_id)

The relational algebra query returns:

A) Course titles

B) Instructor names and course IDs

C) Salaries

D) Departments only

Q8. Which SQL operator corresponds to set difference in relational algebra?

A) UNION

B) INTERSECT

C) EXCEPT

D) DISTINCT

Q9. Which SQL clause groups tuples for aggregation?

A) ORDER BY

B) HAVING

C) GROUP BY

D) JOIN

Q10. In a LEFT OUTER JOIN, tuples from which relation are always preserved?

A) Right relation

B) Left relation

C) Both

D) Only matched tuples

Q11. Relational algebra for names and salaries of instructors in Physics earning > 80000:

- Schema:
  - instructor(ID, name, dept_name, salary)

A) σdept_name='Physics'(πname,salary(instructor>80000))

B) πname,salary(σdept_name='Physics' ∧ salary>80000 (instructor))

C) σsalary>80000(πdept_name='Physics'(instructor))

D) πsalary>80000(σdept_name='Physics'(instructor))

Q12. Which expression performs a natural inner join of r and s?

- Schema:
  - r(A,B)
  - s(B,C)

A) r × s

B) σr.B=s.B(r × s)

C) πA,B,C(σr.B=s.B(r × s))

D) ρr,s

Q13. Find courses offered in Fall 2009 and Spring 2010 (SQL):

- Schema:
  - section(course_id, semester, year)

```
SELECT course_id FROM section WHERE semester='Fall' AND year=2009
_____________________
SELECT course_id FROM section WHERE semester='Spring' AND year=2010;
```

Fill in the blank:

A) UNION

B) EXCEPT

C) INTERSECT

D) JOIN

Q14. Which of the following can be said about the SQL query below intended to "Departments with average salary > 68000 (SQL)":

- Schema:
  - instructor(dept_name, salary)

```
SELECT dept_name, AVG(salary)
FROM instructor
GROUP BY dept_name
HAVING AVG(salary) > 68000;
```

A) Correct

B) Incorrect (must use WHERE for aggregate)

C) Incorrect (no need for GROUP BY)

D) Incorrect (missing table)

Q15. Relational algebra equivalent of SELECT DISTINCT dept_name FROM instructor;

- Schema:
  - instructor(ID, name, dept_name, salary)

A) σdept_name(instructor)

B) πdept_name(instructor)

C) ρdept_name(instructor)

D) σdistinct(instructor)

Q16. Which of the following can be said about the SQL statement below intended to “Delete instructors working in departments in Watson” (SQL):

- Schema:
  - instructor(ID, dept_name)
  - department(dept_name, building)

```
DELETE FROM instructor
WHERE dept_name IN (
  SELECT dept_name FROM department WHERE building='Watson'
);
```

A) Correct

B) Incorrect (deletes department)

C) Incorrect (missing IN)

D) Incorrect (missing WHERE)

Q17. Relational division concept: students who have taken all Biology courses is expressed using:

- Schema:
  - takes(ID, course_id)
  - course(course_id, dept_name)

A) r ÷ s

B) r × s

C) r – s

D) s ÷ r

Q18. SQL equivalent of πname(σdept_name='Comp. Sci.'(instructor ⋈ teaches)):

- Schema:
  - instructor(ID, name, dept_name)
  - teaches(ID, course_id)

```
SELECT name FROM instructor, teaches
WHERE dept_name='Comp. Sci.' AND instructor.ID=teaches.ID;
```

A) Correct

B) Incorrect (missing join condition)

C) Incorrect (no WHERE)

D) Incorrect (wrong table)

Q19. What does SQL do for DELETE with subquery referencing same table?

A) Errors at runtime

B) Deletes recursively

C) Computes subquery first then deletes

D) Loops indefinitely

Q20. Define a foreign key with cascading updates (SQL fragment):

- Schema:
    - course(course_id, dept_name)
    - department(dept_name)

```
FOREIGN KEY (dept_name) REFERENCES department
ON UPDATE CASCADE
```

A) Correct

B) Incorrect (on delete only)

C) Incorrect (missing FOREIGN KEY)

D) Incorrect (PRIMARY KEY used)

Q21. Relational algebra: instructors earning more than every Biology instructor:

- Schema:
  - instructor(name, salary, dept_name)

A) πname(σsalary>some(σdept_name='Biology'(instructor)))

B) πname(σsalary>all(σdept_name='Biology'(instructor)))

C) πname(σsalary>=avg(Biology)(instructor))

D) None

Q22. Which of the following can be said for the query below which attempts to "derive maximum salary"
Schema:

- instructor(salary)

```
SELECT MAX(salary) FROM instructor;
```

A) Correct

B) Incorrect (needs GROUP BY)

C) Incorrect (uses HAVING)

D) Incorrect (use DISTINCT)

Q23. Relational algebra for customers who have a loan and an account:

- Schema:
  - borrower(customer_name, loan_number)
  - depositor(customer_name, account_number)

A) πcustomer_name(borrower) ∩ πcustomer_name(depositor)

B) borrower × depositor

C) borrower − depositor

D) depositor ÷ borrower

Q24. A trigger to convert blank grades to NULL should run:

- Schema:
  - takes(grade)

A) AFTER UPDATE OF grade

B) BEFORE UPDATE OF grade

C) BEFORE INSERT OF grade

D) AFTER DELETE

Q25. Which trigger action updates tot_cred when a passing grade is set?

- Schema:
  - student(id, tot_cred)
  - takes(id, course_id, grade)
  - course(course_id, credits)

A) AFTER UPDATE ON takes(grade)

B) BEFORE UPDATE ON course

C) BEFORE DELETE ON takes

D) AFTER INSERT ON course

Q26. If you INSERT into view faculty(ID,name,dept_name) only, what happens to salary in base instructor?

- Schema:
  - faculty view: (ID, name, dept_name) derived from instructor(ID, name, dept_name,salary)

A) Inserted only into view

B) Inserted into instructor with salary NULL

C) Fails by constraint

D) Updates department

Q27. Relational algebra: customers with accounts at all branches located in Brooklyn:

- Schema:
  - depositor(customer_name, branch_name)
  - account(branch_name, balance)
  - branch(branch_name, branch_city)

A) πcustomer_name,branch_name(depositor × account) ÷ πbranch_name(σbranch_city='Brooklyn'(branch))

B) depositor × branch

C) depositor − account

D) depositor ⋈ branch

Q28. Find the names of employees who earn more than all employees in the HR department.

- Schema:
  - employee(emp_id CHAR(5), emp_name VARCHAR(20), dept_name VARCHAR(20), salary NUMERIC(8,2))

```
SELECT emp_name
FROM employee
WHERE salary ____ (
    SELECT salary
    FROM employee
    WHERE dept_name = 'HR'
);
```

A) > some

B) > all

C) >= all

D) in

Q29. Display product IDs sold in 2024 but not in 2023.

- Schema:
  - sales(product_id CHAR(6), year NUMERIC(4), amount NUMERIC(10,2))

```
SELECT product_id
FROM sales
WHERE year = 2024
AND product_id ____ (
    SELECT product_id
    FROM sales
    WHERE year = 2023
);
```

A) in

B) not in

C) between

D) exists

Q30. Create a view showing each customer's total purchase amount.

- Schema:
  - orders(order_id CHAR(6), cust_id CHAR(5), amount NUMERIC(8,2))
  - customer(cust_id CHAR(5), cust_name VARCHAR(20))

```
CREATE VIEW customer_totals AS
SELECT cust_id, SUM(amount) AS total_spent
FROM orders
____ cust_id;
```

A) where

B) group by

C) having

D) order by

Q31. Using the above view, find customers who spent more than the average of all totals.

- Schema:
  - customer_totals(cust_id CHAR(5), total_spent NUMERIC(10,2))

```
SELECT cust_id
FROM customer_totals
WHERE total_spent > (
   SELECT ____ (total_spent)
   FROM customer_totals
);
```

A) min

B) avg

C) sum

D) count

Q32. List department names whose average employee salary exceeds $60,000.

- Schema:
  - employee(emp_id CHAR(5), emp_name VARCHAR(20), dept_name VARCHAR(20), salary NUMERIC(8,2))

```
SELECT dept_name, AVG(salary)
FROM employee
GROUP BY dept_name
____ AVG(salary) > 60000;
```

A) where

B) having

C) and

D) limit

Q33. Find the employee name and project name for every employee working on at least one project.

- Schema:
  - employee(emp_id CHAR(5), emp_name VARCHAR(20))
  - works_on(emp_id CHAR(5), proj_id CHAR(5))
  - project(proj_id CHAR(5), proj_name VARCHAR(30))

```
SELECT e.emp_name, p.proj_name
FROM employee e ____ works_on w
ON e.emp_id = w.emp_id
JOIN project p
ON w.proj_id = p.proj_id;
```

A) natural inner join

B) cross join

C) left outer join

D) full outer join

Q34. Create a view listing departments that have a higher budget than the average department budget.

- Schema:
  - department(dept_name VARCHAR(20), budget NUMERIC(10,2))

```
Create wiew dept_avg(value) AS (
   SELECT AVG(budget)
   FROM department
)
SELECT dept_name
FROM department, dept_avg
WHERE department.budget ____ dept_avg.value;
```

A) >=

B) >

C) <

D) !=

Q35. Delete all projects whose total cost is less than the average project cost.

- Schema:
  - project(proj_id CHAR(5), proj_name VARCHAR(30), total_cost NUMERIC(10,2))

```
DELETE FROM project
WHERE total_cost < (
   SELECT ____ (total_cost)
   FROM project
);
```

A) avg

B) sum

C) count

D) max

Q36. Create a trigger to automatically set a null commission value to zero after an insert.

- Schema:
  - salesperson(sales_id CHAR(5), name VARCHAR(20), commission NUMERIC(5,2))

```
CREATE TRIGGER set_default_commission
AFTER INSERT ON salesperson
REFERENCING NEW ROW AS nrow
FOR EACH ROW
WHEN (nrow.commission IS ____ )
BEGIN ATOMIC
   UPDATE salesperson
   SET commission = 0
   WHERE sales_id = nrow.sales_id;
END;
```

A) empty

B) null

C) zero

D) missing