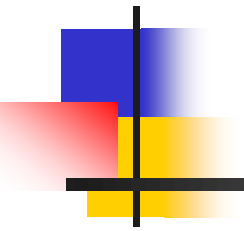
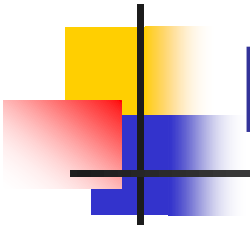
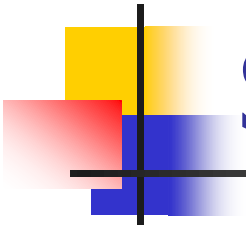


SQL Exercises [Part-2]



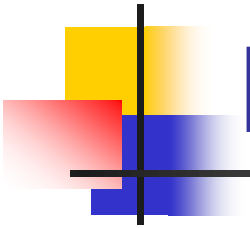
Exercise 1

- A database system should be simple so that many users can interact with the system conveniently.
- It should also be complex so that many queries and transactions can be handled efficiently.
- How is the above achieved?



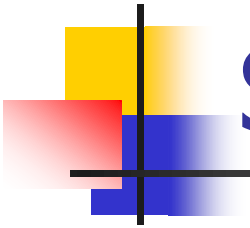
Solution 1

- It is achieved through the layered architecture (i.e. different levels of abstraction)
  - External View describes how user sees the data
  - Conceptual View describes the logical structure of the DB
  - Physical (Internal) View describes the storage structure of data and indices



Exercise 2

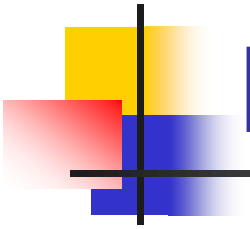
- Consider the following relational schema:
  - Articles(ID, dateline, headline, author, text)
- Use SQL to modify Articles so that any article with NULL text is given the headline of the article as text



Solution 2

- Articles(ID, dateline, headline, author, text)
- Modify Articles so that any article with NULL text is given the headline of the article as text

```
Update Articles
Set text = headline
Where text IS NULL
```

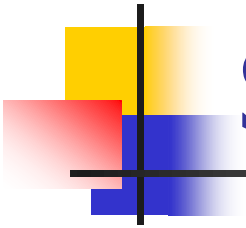


Exercise 3

- **Given the following table:**

- **Product** (product_id, vendor_id, product_price)

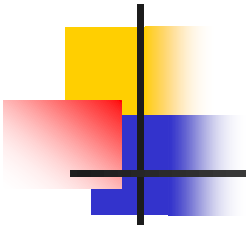
- Use SQL to list all vendors who have 2 or more products priced at 4 or more.



Solution 3

- **Product** (product_id, vendor_id, product_price)
- List all vendors who have 2 or more products priced at 4 or more.

```
Select vendor_id
From Product
Where product_price >=4
Group By vendor_id
Having count(*) >= 2
```

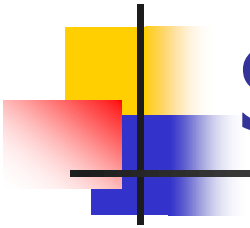


Exercise 4

- We have a relation R(a, b, c), and an SQL query of the form:

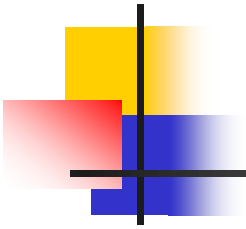
```
SELECT a, avg(b), max(c)
FROM R
WHERE x
GROUP BY a
HAVING y;
```

(a) Can the condition c>50 be used in place of x?



Solution 4(a)

- Yes, the condition can be used in the WHERE clause.
- There are no constraints on the use of attributes in the WHERE clause.

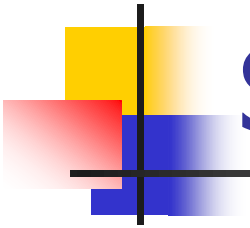


Exercise 4

- We have a relation R(a, b, c), and an SQL query of the form:

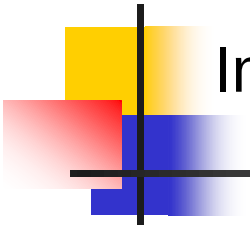
```
SELECT a, avg(b), max(c)
FROM R
WHERE x
GROUP BY a
HAVING y;
```

(b) Can the condition c>50 be used in place of y?



Solution 4(b)

- No, the condition can NOT be used in the HAVING clause
- Only those attributes mentioned in the GROUP BY clause may appear unaggregated in the HAVING clause (Same rule as SELECT clause)



Important points to remember:

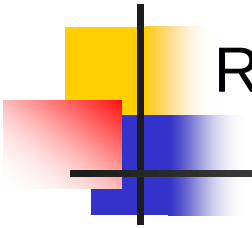
GROUP BY cannot use column aliasing. A GROUP BY clause must contain the column or expressions on which to perform the grouping operation.

Incorrect way:

```
Select deptno as department, count (*) as cnt
From emp
Group by department
```

Correct way is:

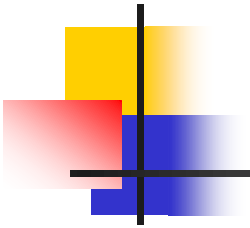
```
Select deptno as department, count (*) as cnt
From emp
Group by deptno
```



Restriction on SELECT Lists with Aggregation

If any aggregation is used, then each element of a SELECT clause must either be aggregated or appear in a group-by clause.

i.e. as a rule, when using GROUP BY and aggregate functions, any items in the SELECT list not used as an argument to an aggregate function must be included in the GROUP BY clause.



What is the difference between the outputs of the following two queries?

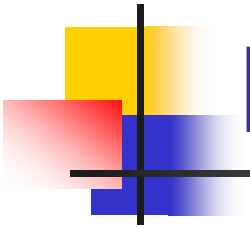
Statement 1:

```
SELECT COUNT (*), SUM (comm)
FROM hr.employees;
```

Statement 2:

```
SELECT COUNT (comm), SUM (comm)
FROM hr.employees;
```

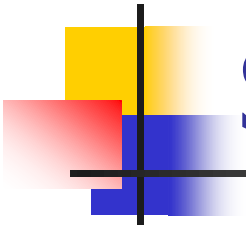
The COUNT (*) will count all rows in the table.
The COUNT (comm) will count only the number commission (comm) values that appear in the table. If there are any rows with a NULL commission, statement 2 will not count them.



Exercise 5

- Consider the following relational schema:
  - **movie**(title, year, length, studio_name, producer_name)

(a) Find the titles that have been used for two or more movies in different years.

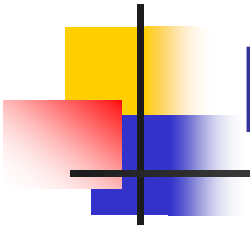


Solution 5(a)

- **movie**(title, year, length, studio_name, producer_name)

Find the titles that have been used for two or more movies in different years.

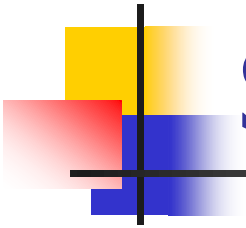
```
SELECT DISTINCT movie.title
FROM movie, movie AS mov
WHERE movie.title = mov.title AND
        movie.year ≠ mov.year
```



Exercise 5

- Consider the following relational schema:
  - **movie**(title, year, length, studio_name, producer_name)

(b) Find the total length of all movies for only those producers who made at least one film prior to 1950



Solution 5(b)

- **movie**(title, year, length, studio_name, producer_name)

Find the total length of all movies for only those producers who made at least one film prior to 1950

```
SELECT producer_name, SUM(length)
FROM movie
GROUP BY producer_name
HAVING MIN(year) < 1950
```