

Command ⇒

⇒ Create View V

as

(select ename, salary, e_mobile
from emp);

⇒ select ename from V;

28/02/2024

Schemas {Collection of Related tables}

- Added with SQL-2.

- By schema name, authorization.

eg → Create Schema Company authorization Rajat.

Catalog

- Named Collection of Schema in a SQL environment.

- SQL environment is a installation of SQL Compliant RDBMS.

- Every catalog has special Schema named as information_

Schema. has information to each schema in that Catalog.

Select -

from -

where -

group by -

having -

order by -

- join

- Sub query
(nested)

- Correlated queries

Correlated queries.

→ There will be a main query and a subquery.
(Outer query) (Inner query)

→ For each row of outer query, inner query will be executed.

Tables →

Bank Account (depositor)

Loan.

query → List of all account holders who have atleast one loan.

⇒ Select acc-name from depositor as D
where ^{exists} (Select * from loan as L where
D.acc-name = L.acc-name)

Subqueries

→ List of Employees whose location starts with 'J';

⇒ select ename from Employee

where location in.

(select location from Posting
where location like 'J%')

Here Inner query executes only once and in
Case of correlated it was executing for each row of outer
query.

Performance

i. Correlated {↓ Lowest}
^
Nested query
^
Join query {↑ Highest}

Triggers

→ Block of code which executes automatically on insert, update, delete of some data.

Types {
① Row Level Triggers
② Statement Level triggers.

Type of Call of triggers →

- | | |
|-------------------|--------------------|
| ① After Insertion | ④ Before Insertion |
| ② After Update | ⑤ Before Update |
| ③ After Deletion | ⑥ Before Deletion. |

Syntax

Create [or Replace] Trigger trigger-name

{Before/after} {insert / update / Delete}

{of column-name} on table-name

[for each row] when (condition)

Begin

- Ssql Statements

End.

[] - optional
& ? - any one

Assertions

⇒ Conditions which must always return 'true'.

Syntax

⇒ Create assertion assertion-name
Check (condition)

Eg → Create assertion balance

Check (balance >= 5000)

SET Clauses

A
C1
a ₁
a ₂
a ₃
a ₄
a ₅
a ₆

B
C1
a ₁
a ₁
a ₃
a ₄
a ₄
a ₅

- Union
- Intersect
- except (minus)

→ Union →

⇒ Select C1 from A Union select C1 from B;

→ No duplicates

C1
a ₁
a ₂
a ₃
a ₄
a ₅
a ₆

⇒ Select C1 from A Union All select C1 from B

C1
a ₁
a ₁
a ₁
a ₂
a ₃
a ₃
a ₄
a ₄
a ₄
a ₅
a ₅
a ₆

→ Intersect →

→ Select C1 from A Intersect Select C1 from B;

C1
a ₁
a ₃
a ₄
a ₅

→ Minus/Except →

→ Select C1 from A except select C1 from B;

C1
a ₂
a ₆

Relational Algebra

→ Procedural Query Language

→ Fundamental Operations

① Selection (σ)

② Projection (π)

③ Union (\cup)

④ Set Difference ($-$)

⑤ Set Intersection (\cap)

⑥ Rename (ρ) ~~(row)~~

⑦ Cartesian Product (\times)

⑧ And (\wedge)

⑨ OR (\vee)

→ Projections are ^{what} ~~the~~ attributes that we want like Select in SQL.

→ Selections are what records we want from the our relation just like where clause in SQL.

① Query → List ^{names} of all employees.

→ $\pi_{ename} (Emp)$

② List name & salary of all employees.

→ $\pi_{ename, salary} (Emp)$

③ → List name & salary of all the employees whose salary is more than 10000;

→ $\pi_{ename, salary} (\sigma_{Salary > 10000} (Emp))$

Emp		
eid	ename	Salary
1	A	
2	B	
3	C	
4	D	
5	E	
6	F	
7	G	

- ④ List name of the employees who have salary more than 10000 & Less 25000;

$\pi_{ename} (\sigma_{salary > 10000 \wedge salary < 25000} (Emp))$

- ⑤ Salary more than 10000 or Less than 25000

$\pi_{ename} (\sigma_{salary > 10000 \vee salary < 25000} (Emp))$

empid	ename	sal
1	A	10000
2	B	15000
3	C	20000
4	D	25000
5	E	30000
6	F	35000
7	G	40000