Database Systems

Lecture

Views

Contents

- Views overview
- Creating views
- Querying views
- Limitations of views
- Modifying base tables through views
- Deleting Views

Views

- A table derived from other table(s)
  - The ‘other’ tables are called base tables
  - A view can also be created from other views…
- Views may be considered as virtual tables
  - They don’t need to physically exist on the disk
- Typical use case is to create a view of join of multiple tables
- Since views are created using ‘select’ queries, therefore they are normally used to make complex queries easier!
- Another use case of views is security
  - Different users can view different views of the same base table!

Views

- Whenever the base table(s) is/are updated, the view also gets updated
- Since views are also tables, therefore we can perform retrieval queries on the views just like base tables
  - But use of insert / update / delete queries on views is tricky…

Creating Views

```
CREATE VIEW   view_name [(v_col1, v_col2, ...)]
AS
SELECT    select_statement;
```

This ‘select’ query is called the ‘defining query’ for the view

It may contain join of multiple tables, aggregate functions, anything!

Creating Views

```
CREATE VIEW    view_name [(v_col1, v_col2, ...)]
AS
SELECT    select_statement;
```

Optional – only required if we need to rename the original columns from the table

If this list is not provided, then the names of columns in the view will be the same as in base table

If this list is provided, it should contain same number of columns as returned by the defining query

Creating Views

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Salary	Super_ssn	Dno

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date

DEPT_LOCATIONS

Dnumber	Dlocation

PROJECT

Pname	Pnumber	Plocation	Dnum

WORKS_ON

Essn	Pno	Hours

Creating Views

```
CREATE VIEW     WORKS_ON1
AS SELECT       Fname, Lname, Pname, Hours
    FROM        EMPLOYEE, PROJECT, WORKS_ON
    WHERE       Ssn = Essn AND Pno = Pnumber;

CREATE VIEW     DEPT_INFO(Dept_name, No_of_emps, Total_sal)
AS SELECT       Dname, COUNT (*), SUM (Salary)
    FROM        DEPARTMENT, EMPLOYEE
    WHERE       Dnumber = Dno
    GROUP BY    Dname;
```

WORKS_ON1

Fname	Lname	Pname	Hours

DEPT_INFO

Dept_name	No_of_emps	Total_sal

Creating Views

```
CREATE VIEW   view_name [(v_col1, v_col2, ...)]
AS
SELECT   select_statement;
```

Example:

```
CREATE VIEW   student_view (Reg, CGPA)
AS
SELECT   S_Reg, S_CGPA
FROM          Student
WHERE    S_CGPA > 3.7;
```

Create a view of only those students which are to be considered for medal

Creating Views

```
CREATE VIEW   view_name [(v_col1, v_col2, ...)]
AS
SELECT    select_statement;
```

Example:

```
CREATE VIEW   production_view
AS
SELECT    *
FROM          Employee
WHERE     Dept_Name = 'Production';
```

Create a view of only those employees who work in the 'production' department

This view may be made accessible only to the head of the production department!

Querying Views

```
SELECT    Reg, MAX (CGPA)
FROM           student_view;

SELECT    Reg, CGPA
FROM           student_view
ORDER BY  CGPA;

SELECT    Name, Salary
FROM           production_view
WHERE          Salary < AVG (Salary);
```

We use the column names from the view, NOT from the table!

Limitations of Views

- A view having ‘group by’ clause in its defining query cannot be joined with another table or view
- Modifying a view modifies the base table!
  - View is just a snapshot – it’s not the actual table
- Modifications in views are not always possible!
- A view can only be modified if
  - It is created using a single base table
  - The defining query does not use ‘Distinct’ keyword
  - No aggregate functions are used in defining query
  - Modification does not violate any constraints in the base table

Modifying Base Table Through Views

- Remember: A view generally shows us a subset of rows from the base table
- What would happen if we execute an insert query that adds data to the view?
- The query may add rows that should not be visible within the view!
- We'll see this through an example…

Modifying Base Table Through Views

- Let’s assume we make a view from Employee table to see only those employees who have VP in their job title

Modifying Base Table Through Views

```
CREATE  vps AS
    SELECT
        employeeNumber,
        lastname,
        firstname,
        jobtitle,
        extension,
        email,
        officeCode,
        reportsTo
    FROM
        employees
    WHERE
        jobTitle LIKE '%VP%';
```

Modifying Base Table Through Views

- Querying the view will show us something like this:

```
SELECT * FROM vps;
```

employeeNumber	lastname	firstname	jobtitle
1056	Phan	Mary	VP Sales
1076	Firrelli	Jeff	VP Marketing

Modifying Base Table Through Views

- Now let’s try to add a row in the base table through the view

Modifying Base Table Through Views

```
INSERT INTO vps(
    employeeNumber,
    firstName,
    lastName,
    jobTitle,
    extension,
    email,
    officeCode,
    reportsTo
)
VALUES(
    1703,
    'Lily',
    'Bush',
    'IT Manager',
    'x9111',
    'lilybush@classicmodelcars.com',
    1,
    1002
);
```

Modifying Base Table Through Views

- We just added an IT manager to the Employee table
- But this is not what we want from views!
- The view was only supposed to expose those employees whose job title includes ‘VP’, not other employees
- We need to ensure the consistency of a view so that only those rows are inserted/updated/deleted which are visible through the view
- Thus is done using WITH CHECK OPTION

Modifying Base Table Through Views

```
CREATE VIEW   view_name
AS
SELECT   select_statement
WITH CHECK OPTION;
```

- If we use WITH CHECK OPTION, the DBMS rejects any queries on the view that change the rows that should not be visible in the view.

Modifying Base Table Through Views

```
CREATE VIEW CUSTOMERS_VIEW
AS
SELECT name, age
FROM  CUSTOMERS
WHERE age IS NOT NULL
WITH CHECK OPTION;

INSERT INTO CUSTOMERS_VIEW(name,age)
VALUES (“Ali”, null);
```

Error Code: 1369. CHECK OPTION failed 'CUSTOMERS_VIEW'

Deleting Views

- Just like we remove tables, columns, constraints, etc., we can remove a view by using the DROP command.

```
DROP VIEW      view_name [RESTRICT|CASCADE]
```

- RESTRICT will prevent deleting a view if other views are dependent on it
- CASCADE will delete this view and all other views that are dependent on it

Views as Authorization Mechanisms

- views can be used to hide certain attributes or tuples from unauthorized users.

```
CREATE VIEW   DEPT5EMP   AS
SELECT        *
FROM          EMPLOYEE
WHERE         Dno = 5;
```

```
CREATE VIEW   BASIC_EMP_DATA   AS
SELECT        Fname, Lname, Address
FROM          EMPLOYEE;
```

Thanks a lot