VIEWS

CIS-673, LECTURE#18

BY RAJ PATIL

2 VIEWS

- Virtual Table
- A view is a logical representation of one or more tables.
- A view is a stored query.
- A view derives its data from the tables on which it is based, called **base tables**
- All operations performed on a view affect the base tables.

VIEW EXAMPLE – HIDE METADATA

```
create table emp
(
Employee_ID int primary key,
Employee_Name varchar(20),
Job_Title varchar(20),
Salary number(8,2),
Gender varchar(1),
Marital_Status varchar(10),
Dept_ID int references dept(Dept_ID)
);
```

```
CREATE VIEW empview AS
SELECT Employee_ID as eid,
Employee_Name as ename,
Job_Title as designation,
Salary as income
FROM emp;
```

- Select * from empview;

BENEFITS

- **Security:** by restricting access to a set of rows or columns of a table
- **Hide data complexity:** a single view can be defined with a join, which is a collection of related columns or rows in multiple tables.
- Table and columns of a view can be renamed without affecting the tables on which the view is based.
- Isolate applications from changes in definitions of base tables

5 CHARACTERISTICS

- Unlike a table, a view does not contain any data.
- A view is defined by a query that extracts or derives data from the base tables referenced by the view.
- Because a view is based on other objects, it requires no storage other than storage for the query.
- Views provide a different representation (such as subsets or supersets) of the data that resides within other tables and views.
- Each time a view is referenced, its associated select-query is executed that represents the view.

VIEW OPTIONS

- Default (writable)
- Read only
- With check option
  - INSERT and UPDATE statements issued on the view cannot result in rows that the view cannot select later.

DEFAULT OPTION

- CREATE VIEW view_one AS

  SELECT Employee_ID, Employee_Name, Job_Title, Salary

  FROM employee;

- Add record: insert into view_one values(7584, 'Mark','consultant',40000.00);
- Raise the salary: update viewone set Salary=83000 where Employee_ID=815;
- Delete record: delete from viewone where Employee_ID = 555;

- Notice that the Inserts, Updates, Deletes **affect the Base Table**.

READ OPTION

- CREATE VIEW viewtwo AS

  SELECT Employee_ID, Employee_Name, Job_Title, Salary

  FROM emp WITH READ ONLY ;

- For read only view: No inserts, no updates, no deletes.

- The following statements throw error:
  - insert into viewtwo values(608, 'Mike','Business Analayst',70000);
  - update viewtwo set Salary=81000 where Employee_ID=815;
  - delete from viewtwo where Employee_ID = 815;

WITH CHECK OPTION

- With check option: Preserves the where condition of the view
- CHECK OPTION creates the view with the constraint so that the INSERT and UPDATE statements issued on the view cannot result in rows that the view cannot select later.

- CREATE VIEW view_three AS

  SELECT empno, ename, job_type, salary, deptno

  FROM emp **WHERE deptno = 40**

  WITH CHECK OPTION CONSTRAINT it_cnst;

- Following statements result in violation of where clause, and therefore result in error:
  - INSERT INTO view_three VALUES (7591, 'William', 'Web designer',95000,20);
  - update view_three set Dept_ID=20 where Employee_Name='Matt';

JOIN VIEW EXAMPLE

- CREATE VIEW empdept AS

  SELECT emp.Employee_ID, emp.Employee_Name, dept.Dept_ID, dept.Dept_Name , dept.Manager_Name

  FROM emp, dept

  where emp.Dept_ID = dept.Dept_ID;

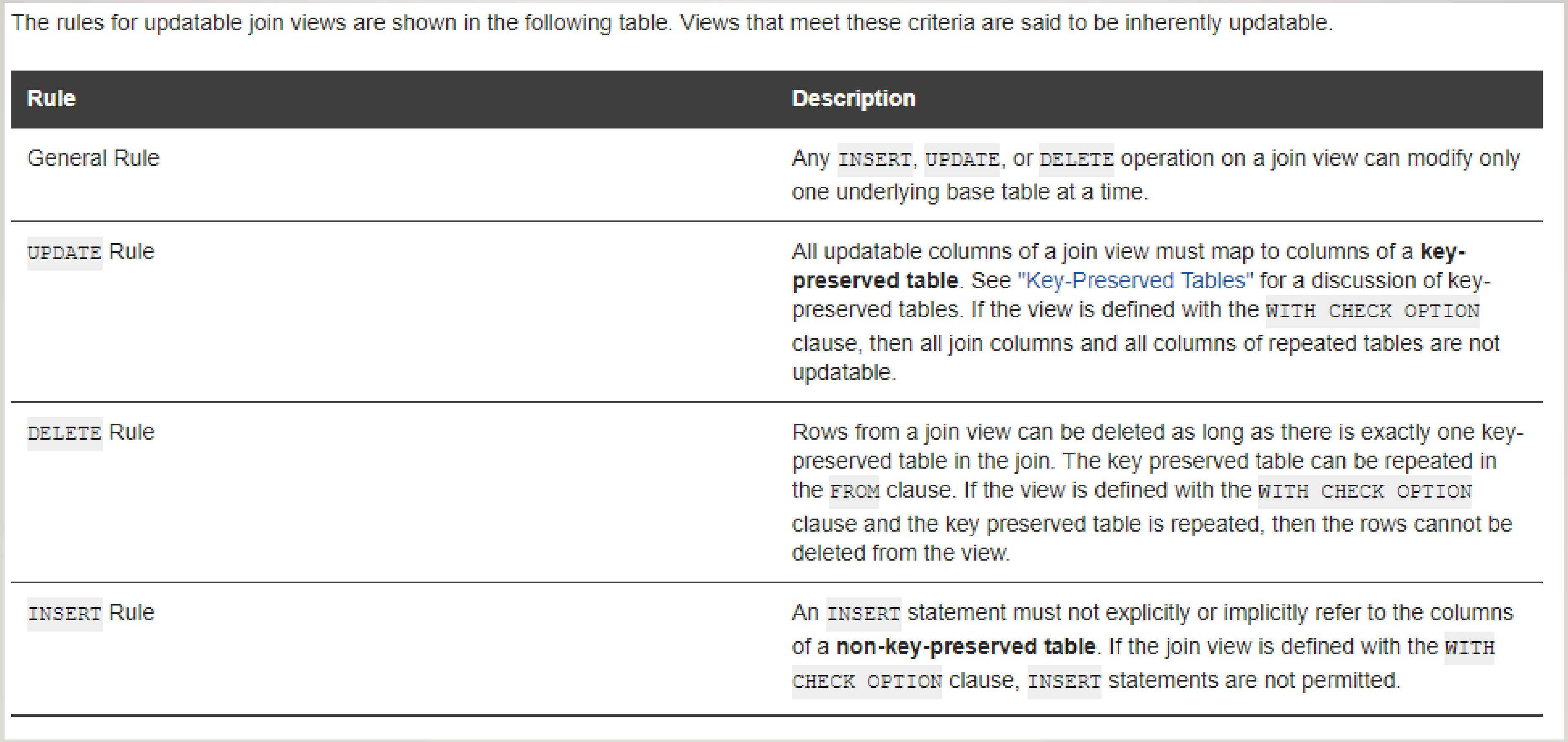
- Outcome of DML operations on Join-View depends upon Key-Preserved Table.

- A base-table is key-preserved if the key of base-table can also be a key of the result of the join.

- For example, emp is a key-preserved table, because Employee_ID is a key of the emp table, and also a key of the result of the join.

- dept is not a key-preserved table, because although Dept_ID is a key of the dept table, it is not a key of the join.

JOIN VIEW - RULES



The rules for updatable join views are shown in the following table. Views that meet these criteria are said to be inherently updatable.

Rule	Description
General Rule	Any `INSERT`, `UPDATE`, or `DELETE` operation on a join view can modify only one underlying base table at a time.
`UPDATE` Rule	All updatable columns of a join view must map to columns of a **key-preserved table**. See "Key-Preserved Tables" for a discussion of key-preserved tables. If the view is defined with the `WITH CHECK OPTION` clause, then all join columns and all columns of repeated tables are not updatable.
`DELETE` Rule	Rows from a join view can be deleted as long as there is exactly one key-preserved table in the join. The key preserved table can be repeated in the `FROM` clause. If the view is defined with the `WITH CHECK OPTION` clause and the key preserved table is repeated, then the rows cannot be deleted from the view.
`INSERT` Rule	An `INSERT` statement must not explicitly or implicitly refer to the columns of a **non-key-preserved table**. If the join view is defined with the `WITH CHECK OPTION` clause, `INSERT` statements are not permitted.

DML ON JOIN VIEW

- Any INSERT, UPDATE, or DELETE operation on a join view can modify only one underlying base table at a time.
- Insert, update, delete on join views work as long as they impact only the key-preserved table.
- DML operations cannot impact or be performed on the NON key-preserved table.

INSERT – JOIN VIEW

- insert into empdept (Employee_ID, Employee_Name, Salary) values(720, 'Mark',76000);
    - Success, since it impacts/modifies only the key-preserving table

- insert into empdept (Dept_ID, DEPT_NAME, MANAGER_NAME) values(50, 'Production','James');
    - Error, since attempt to modify non-key preserving table

- insert into empdept (Employee_ID,Dept_ID, DEPT_NAME) values(770, 10,'Sales');
    - Error, since attempt to modify more than one base table

UPDATE – JOIN VIEW

- All updatable columns of a join view must map to columns of a key-preserved table.

- Examples:

  - UPDATE empdept SET salary = salary * 1.10 WHERE Dept_ID = 20; -- success

  - UPDATE empdept SET Dept_Name = 'QA' WHERE Dept_ID = 10; --error

  - UPDATE empdept SET Dept_ID = 200 where employee_id = 449; --error

DELETE – JOIN VIEW

- You can delete from a join view provided there is *one and only one* key-preserved table in the join.

- Delete statement affects only the key-preserved table, and not the non-key preserved table.
  - delete from empdept where Dept_ID=10;
  - delete from empdept where Employee_ID=449;
  - delete from empdept where Dept_Name='Quality Assurance' or 'Information Technology';

- The above DELETE statements on the emp_dept view are successful because they can be translated to a DELETE operation on the base emp table, and because the emp table is the only key-preserved table in the join.

- delete always end up removing the rows from the key-preserved table

RIGHT JOIN VIEW - EXAMPLE

- insert into dept values(500,'HR',400,'Bob');

- CREATE VIEW rightjoinview AS

  SELECT emp.Employee_ID, emp.Employee_Name, emp.Salary, dept.Dept_ID, dept.Dept_Name, dept.Manager_Name

  from emp right JOIN dept ON emp.Dept_ID = dept.Dept_ID;

- All the following DML operations (which were successful on natural join view), fail here because there is no key preserving table:
  - insert into rightjoinview (Employee_ID, Employee_Name, Salary) values(720, 'Mark',76000);
  - update rightjoinview SET Employee_id = 210 WHERE Manager_Name = 'Bob';
  - delete from rightjoinview where Employee_ID=230;

LEFT JOIN VIEW - EXAMPLE

- insert into emp values(543,'Dave','DBA',90000.00,'M','Single',50); --drop referential-integrity constraint and insert the following row
- CREATE VIEW leftjoinview AS

  SELECT emp.Employee_ID, emp.Employee_Name, emp.Salary,

  dept.Dept_ID, dept.Dept_Name, dept.Manager_Name

  from emp LEFT JOIN dept ON emp.Dept_ID = dept.Dept_ID;
- All the following DML operations are successful because there is a key preserving table:
  - insert into leftjoinview (Employee_ID, Employee_Name, Salary) values(720, 'Mark',76000);
  - update leftjoinview SET Employee_id = 210 WHERE Manager_Name = ‘Vince’;
  - delete from leftjoinview where Employee_ID=230;

DROP VIEW

- Drop view <viewname>;

- Drop view empdept;