BITS, PILANI – K. K. BIRLA GOA CAMPUS

Database Systems

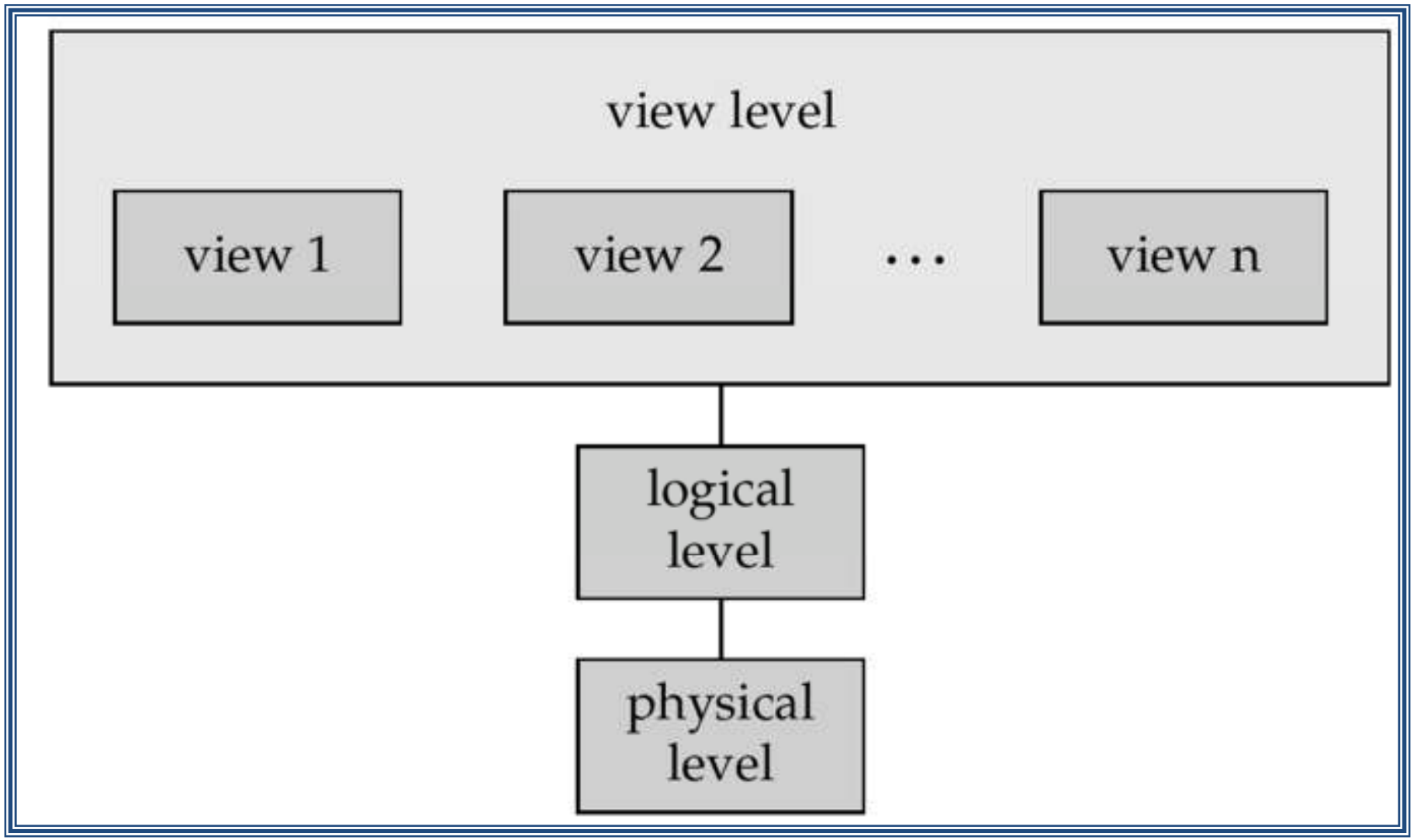
(CSF212)

by

Dr. Mrs. Shubhangi Gawali

Dept. of CS and IS

View of Data



Views

- In some cases, it is not desirable for all users to see the entire logical model (i.e., all the actual relations stored in the database.)
- Consider a person who needs to know a customer's loan number but has no need to see the loan amount. This person should see a relation described, in the relational algebra, by

  $\Pi_{customer\text{-}name,\ loan\text{-}number}\ (borrower \bowtie loan)$
- Any relation that is not of the conceptual model but is made visible to a user as a "virtual relation" is called a **view**.

View Definition

- A view is defined using the **create view** statement which has the form

  create view *v* **as** <query expression>

  where <query expression> is any legal relational algebra query expression. The view name is represented by *v.*
- Once a view is defined, the view name can be used to refer to the virtual relation that the view generates.
- View definition is not the same as creating a new relation by evaluating the query expression
  - Rather, a view definition causes the saving of an expression; the expression is substituted into queries using the view.

Views

To provide a mechanism to hide certain data from the view of certain users.

Eg. Create view all-cust as

(select branchname, custname

From depositor, account

Where depositor.account no = account.account no)

union

(select branchname, custname

From borrower, loan

Where borrower.loan no = loan.loan no)

Find all customers of Dadar branch

Select custname
From all-cust
Where branch-name = ‘Dadar’;

Any updates made in views are reflected in its base table also.

This is the diff betn rename and view.

View Examples

- Consider the view (named *all-customer*) consisting of branches and their customers.

create view *all-customer* **as**

$$\Pi_{\textit{branch-name, customer-name}} (\textit{depositor} \bowtie \textit{account}) \cup \Pi_{\textit{branch-name, customer-name}} (\textit{borrower} \bowtie \textit{loan})$$

We can find all customers of the Zuarinagar branch by writing:

$$\Pi_{\textit{branch-name}} (\sigma_{\textit{branch-name} = \text{“Zuarinagar”}} (\textit{all-customer}))$$

Updates Through View

- Database modifications expressed as views must be translated to modifications of the actual relations in the database.
- Consider the person who needs to see all loan data in the *loan* relation except *amount.* The view given to the person, *branch-loan,* is defined as:

create view *branch-loan* **as**

$$\Pi_{\textit{branch-name, loan-number}} \textit{(loan)}$$

- Since we allow a view name to appear wherever a relation name is allowed, the person may write:

$$\textit{branch-loan} \leftarrow \textit{branch-loan} \cup \{(\text{“Zuarinagar”, L-37})\}$$

Updates Through Views (Cont.)

- The previous insertion must be represented by an insertion into the actual relation *loan* from which the view *branch-loan* is constructed.
- An insertion into *loan* requires a value for *amount*. The insertion can be dealt with by either.
  - rejecting the insertion and returning an error message to the user.
  - inserting a tuple (“L-37”, “Sancoale”, *null*) into the *loan* relation
- Some updates through views are impossible to translate into database relation updates
  - create view v as $\sigma_{branch\text{-}name = \text{“Sancoale”}}$ (*account*))

    $v \leftarrow v \cup$ (L-99, Madgaon, 23)
- Others cannot be translated uniquely
  - *all-customer* $\leftarrow$ *all-customer* $\cup$ {(“Zuarinagar”, “John”)}
    - Have to choose loan or account, and create a new loan/account number!