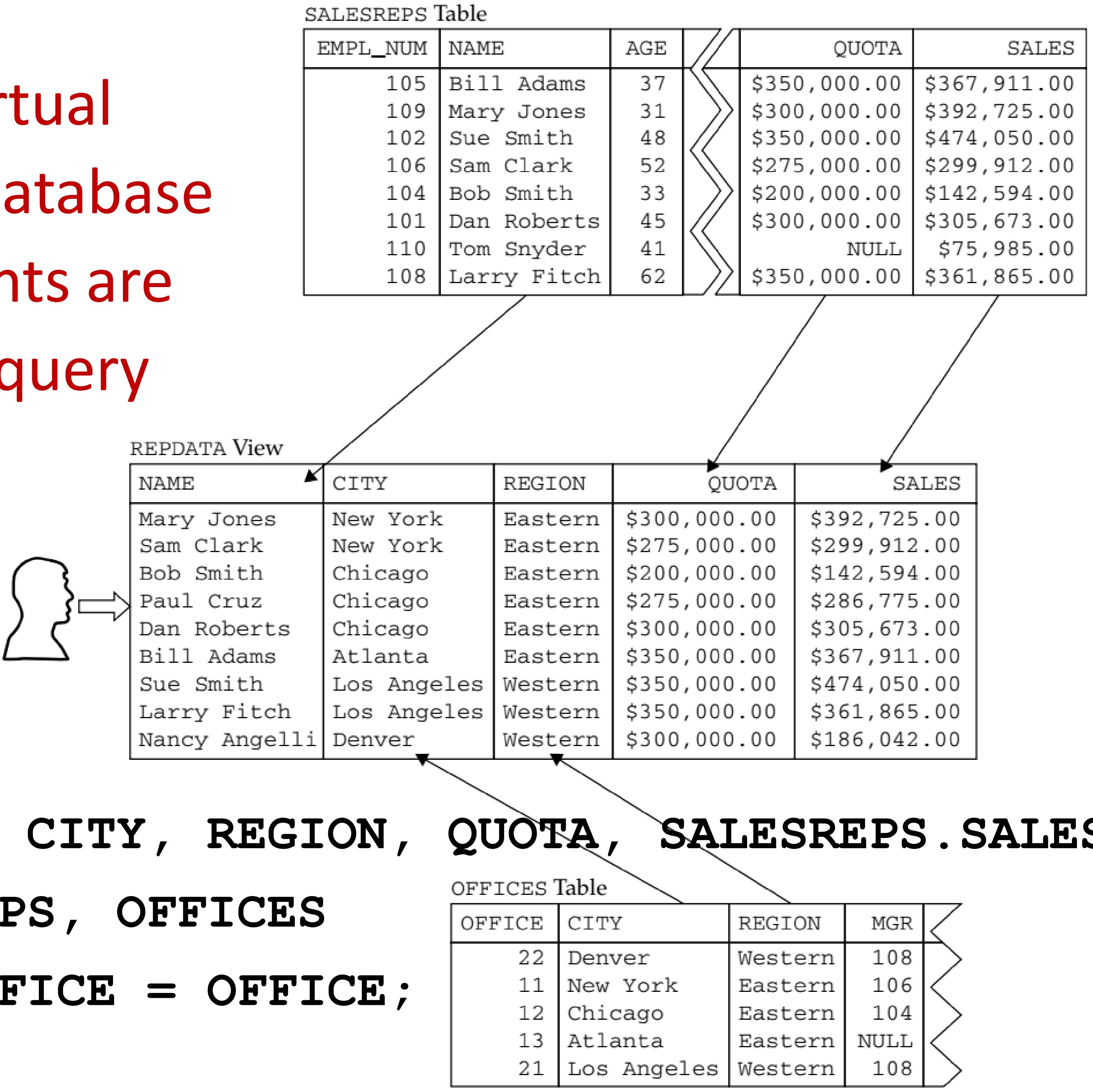
VIEWS

October 2023

Base Tables and Views

- Base Table: A relation whose value is really stored in the database
- View: A relation that is defined in terms of the contents of other tables and views.
  - A “virtual table”, which appears to user just like a real table
  - Unlike real table, it does not exist in DB as a stored set of data values
- SQL creates an illusion of the view by giving the view a name like a table name and storing the definition of the view in the DB

A *view* is a virtual table in the database whose contents are defined by a query



SALESREPS Table

EMPL_NUM	NAME	AGE	QUOTA	SALES
105	Bill Adams	37	$350,000.00	$367,911.00
109	Mary Jones	31	$300,000.00	$392,725.00
102	Sue Smith	48	$350,000.00	$474,050.00
106	Sam Clark	52	$275,000.00	$299,912.00
104	Bob Smith	33	$200,000.00	$142,594.00
101	Dan Roberts	45	$300,000.00	$305,673.00
110	Tom Snyder	41	NULL	$75,985.00
108	Larry Fitch	62	$350,000.00	$361,865.00

REPDATA View

NAME	CITY	REGION	QUOTA	SALES
Mary Jones	New York	Eastern	$300,000.00	$392,725.00
Sam Clark	New York	Eastern	$275,000.00	$299,912.00
Bob Smith	Chicago	Eastern	$200,000.00	$142,594.00
Paul Cruz	Chicago	Eastern	$275,000.00	$286,775.00
Dan Roberts	Chicago	Eastern	$300,000.00	$305,673.00
Bill Adams	Atlanta	Eastern	$350,000.00	$367,911.00
Sue Smith	Los Angeles	Western	$350,000.00	$474,050.00
Larry Fitch	Los Angeles	Western	$350,000.00	$361,865.00
Nancy Angelli	Denver	Western	$300,000.00	$186,042.00

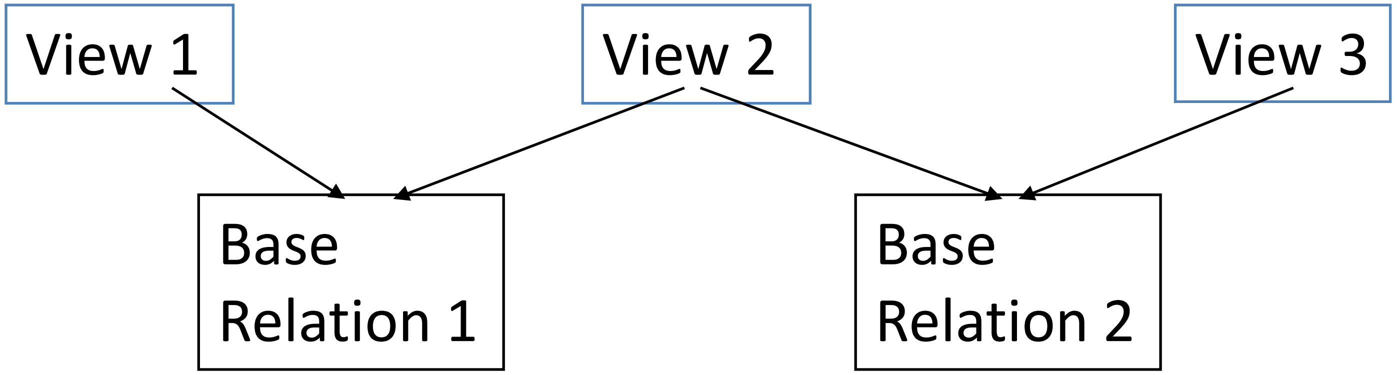
```
SELECT NAME, CITY, REGION, QUOTA, SALESREPS.SALES
FROM SALESREPS, OFFICES
WHERE REP_OFFICE = OFFICE;
```

OFFICES Table

OFFICE	CITY	REGION	MGR
22	Denver	Western	108
11	New York	Eastern	106
12	Chicago	Eastern	104
13	Atlanta	Eastern	NULL
21	Los Angeles	Western	108

Relational Views

- Relations derived from other relations.
- Views have no stored tuples.
- Are useful to provide multiple user views.
- A view may be derived from multiple base relations
- Queries on views are expanded into queries on their base relations.



What Happens When a View is Used?

- A reference to a view is made
  - DBMS finds its definition in the DB
  - DBMS translates the request into an equivalent request against the source table of the view
- Simple views:
  - DBMS construct each row of the view on the fly, drawing the data for the row from the source table
- Complex views:
  - Carry outs the query and stores its result in a temporary table
  - Fills your request for view access from this temporary table and discards the table when it is no longer used

View Creation

```
CREATE VIEW view-name [ ( attr [ , attr ] ...) ]
AS  query
[ WITH CHECK OPTION ] ;
```

```
DROP  VIEW  view-name [CASCADE | RESTRICT];
```

- The query cannot include either UNION or ORDER BY
- You can think of a view as a stored query

Views – Some Examples

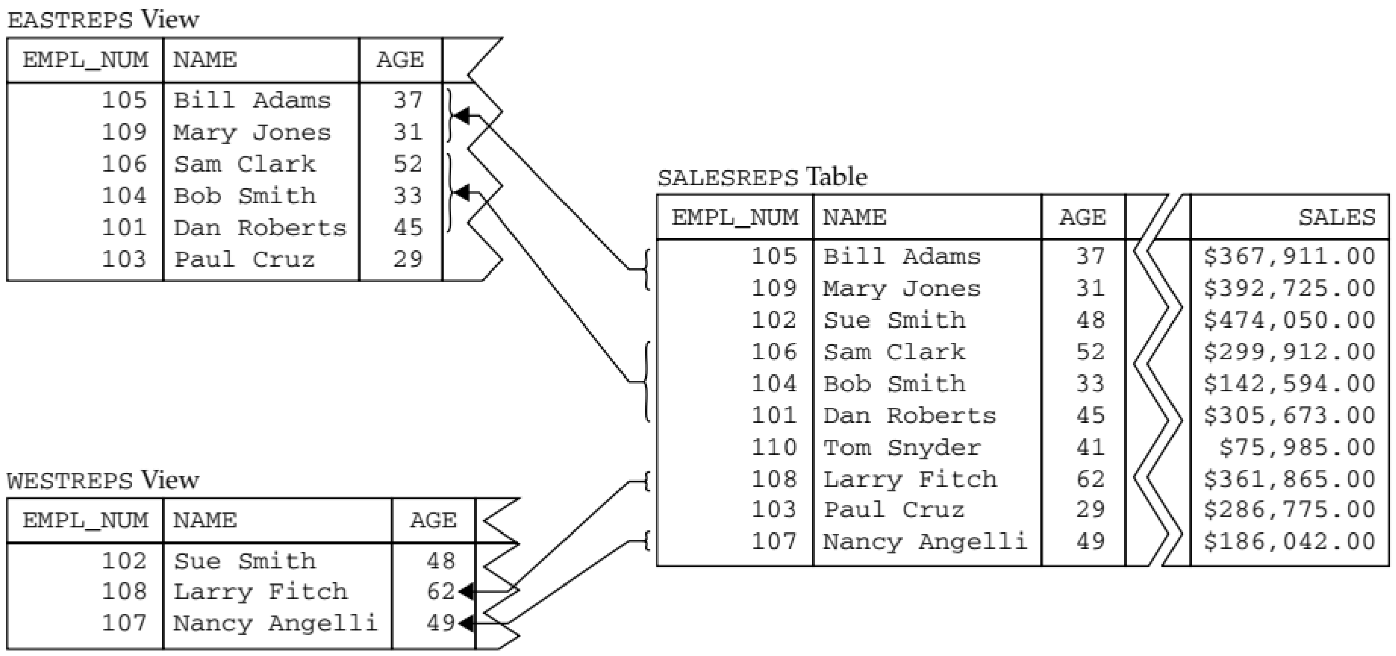
1. Create a view showing eastern region salespeople
2. Create a view of the ‘offices’ table for the order-processing staff that includes the office’s city, office no., and region
3. Define a view that contains summary order data for each salesperson
4. Create a view to get sales and quota information of salespersons along with the city and region of the offices where they work
5. Use the previous view to list the salespeople who are over quota, showing the name, city, and region for each salesperson

Example: View Definition (1)

- Create a view showing eastern region salespeople

```
CREATE VIEW eastreps AS
   SELECT *
   FROM salesreps
   WHERE Rep_office IN (11, 12, 13);
```

Horizontal View



EASTREPS View

EMPL_NUM	NAME	AGE
105	Bill Adams	37
109	Mary Jones	31
106	Sam Clark	52
104	Bob Smith	33
101	Dan Roberts	45
103	Paul Cruz	29

SALESREPS Table

EMPL_NUM	NAME	AGE	SALES
105	Bill Adams	37	$367,911.00
109	Mary Jones	31	$392,725.00
102	Sue Smith	48	$474,050.00
106	Sam Clark	52	$299,912.00
104	Bob Smith	33	$142,594.00
101	Dan Roberts	45	$305,673.00
110	Tom Snyder	41	$75,985.00
108	Larry Fitch	62	$361,865.00
103	Paul Cruz	29	$286,775.00
107	Nancy Angelli	49	$186,042.00

WESTREPS View

EMPL_NUM	NAME	AGE
102	Sue Smith	48
108	Larry Fitch	62
107	Nancy Angelli	49

Example: View Definition (1)

- Create a view of the ‘offices’ table for the staff that includes the office’s city, office no., and region

```
CREATE VIEW officeinfo AS
  SELECT office,city,region
  FROM offices;
```

Vertical View

Vertical View: another example

REPINFO View

EMPL_NUM	NAME	REP_OFFICE
105	Bill Adams	13
109	Mary Jones	11
102	Sue Smith	21
106	Sam Clark	11
104	Bob Smith	12
101	Dan Roberts	12
110	Tom Snyder	NULL
108	Larry Fitch	21
103	Paul Cruz	12
107	Nancy Angelli	22

```
CREATE VIEW REPINFO AS
SELECT EMPL_NUM,NAME,REP_OFFICE
FROM SALESREPS;
```

SALESREPS Table

EMPL_NUM	NAME	AGE	REP_OFFICE	TITLE	HIRE_DATE	MANAGER	QUOTA	SALES
105	Bill Adams	37	13	Sales Rep	2006-02-12	104	$350,000.00	$367,911.00
109	Mary Jones	31	11	Sales Rep	2007-10-12	106	$300,000.00	$392,725.00
102	Sue Smith	48	21	Sales Rep	2004-12-10	108	$350,000.00	$474,050.00
106	Sam Clark	52	11	VP Sales	2006-06-14	NULL	$275,000.00	$299,912.00
104	Bob Smith	33	12	Sales Mgr	2005-05-19	106	$200,000.00	$142,594.00
101	Dan Roberts	45	12	Sales Rep	2004-10-20	104	$300,000.00	$305,673.00
110	Tom Snyder	41	NULL	Sales Rep	2008-01-13	101	NULL	$75,985.00
108	Larry Fitch	62	21	Sales Mgr	2007-10-12	106	$350,000,00	$361,865.00
103	Paul Cruz	29	12	Sales Rep	2005-03-01	104	$275,000.00	$286,775.00
107	Nancy Angelli	49	22	Sales Rep	2006-11-14	108	$300,000.00	$186,042.00

Example: View Definition (2)

- Create a view that contains the customer number, company name, and credit limit of all customers assigned to Amit (emp no. = 105)

```
CREATE VIEW billcust AS
  SELECT custnum, company, creditlimit
  FROM customers
  WHERE custrep = 105;
```

Remember: SQL does not restrict you to purely horizontal/vertical slices of a table

Example: View Definition (3)

- Create a view to get sales and quota information of salespersons along with the city and region of the offices where they work.

```
CREATE VIEW repdata AS
  SELECT name,city,region,quota,salesreps.sales
  FROM salesreps,offices
  WHERE Rep_office = office;
```

You may query a view as if it were a base table.

- Use this view to list the salespeople who are over quota, showing the name, city, and region for each salesperson.

```
    SELECT name,city,region
    FROM repdata
    WHERE sales > quota;
```

Example: View Definition (4)

- Define a view that contains summary order data for each salesperson

```
CREATE VIEW order_by_rep
     (who,howmany,total,low,high,average) AS
  SELECT rep,COUNT(*),SUM(amount), MIN(amount),
                     MAX(amount), AVG(amount)
  FROM orders
  GROUP BY rep;
```

Now show the summary order data (i.e., no. of orders, total order amount, and average order size) for each salesperson with his name

Example: View Definition (4)

- Show the summary order data (i.e., no. of orders, total order amount, and average order size) for each salesperson with his name

```
SELECT name,howmany,total,average
FROM salesreps,order_by_rep
WHERE who = Empl_num
ORDER BY total DESC;
```

Updating a View

- Consider the situation:
  - Insert/Delete/Update a row of data into/from/of a view
- Update on a view actually changes its base relation(s)!
  - Why are some views not updateable?
  - What type of views are updateable?
- Two extreme examples:
  - eastreps view (updatable)
  - ord_by_rep view (not updatable but read only view)

Updating a View

- A view can be updated if the query that defines the view meets these restrictions (ANSI/ISO standard):
  - DISTINCT must not be specified
  - Only 1 source table/view in FROM clause for which user has the required privileges
  - Simple column reference (no expressions, calculated column, or column function) in SELECT clause
  - No subquery in WHERE clause
  - No GROUP BY or a HAVING class
- For the view to be updatable (it is easy to remember):
  - The DBMS must be able to trace any row/column of the view back to its source row in the source table

View - WITH CHECK OPTION

- Enforces the query condition for insertion or update

```
CREATE VIEW eastreps AS
    SELECT *
    FROM salesreps
    WHERE Rep_office IN (11, 12, 13);
```

```
INSERT INTO eastreps (emplnum,name,rep_office,age,sales)
      VALUES (113,'Reena',11,43,0.00)
```

```
INSERT INTO eastreps (emplnum, name,rep_office,age,sales)
      VALUES (114,'Rohit',21,47,0.00)
```

- Integrity checking - create the view with check option

View - WITH CHECK OPTION

```
CREATE VIEW eastreps AS
    SELECT *
    FROM salesreps
    WHERE Rep_office IN (11, 12, 13);
WITH CASCADED/LOCAL CHECK OPTION
```

- If the inserted/modified row would not meet the condition, the INSERT or UPDATE statement fails, and the operation is not carried out
- Options: CASCADED (default) or LOCAL
- Plays an imp role to ensure the integrity of the DB

Materialized Views

- Query processing finished – DBMS discards the temporary tables
- View defined as “materialized view”
- Materialized view:
  - DBMS carry outs the query that defines the view once
  - Stores the results within the DB, and then
  - Permanently maintains this copy of the view data
- Materializing the view contents – very high overhead operation

Materialized Views

- DBMS must automatically examine every change to the data in the underlying source tables and make the corresponding changes in the materialized view data
- When query is to be processed against this view, it has the data already at hand and can process the query very efficiently
- Trade-off between the efficiency of updates on the data contained in the view and the efficiency of queries on the view data
- Most useful when the volume of updates to the underlying data is relatively small, and the volume of queries against the view is relatively high

Pros and Cons

- Advantages:
  - Security
  - Query simplicity
  - Structural simplicity
  - Insulation from change
  - Data integrity
- Disadvantages:
  - Performance
  - Update restrictions