

SQL FOR DB2

Chapter 11

Views and Indexes

CHAPTER OBJECTIVES

- ① Create views
- ① Create indexes

WHAT IS A VIEW?

- ① A view is an object that defines an access path to data stored in one or more tables
- ① Views can be used to:
 - ① Sequence data in a table in a different order
 - ① Select or omit specific rows from tables
 - ① Select or omit specific columns within rows from tables
 - ① Access data from two or more tables

CREATING A VIEW

- ① To define a view, the `CREATE VIEW` keywords are specified, followed by the name of the view
- ① A view must not have the same name as any object in the same schema
- ① The view name is followed by the keyword `AS`, and the actual subquery follows `AS`

COMPONENTS OF THE CREATE VIEW COMMAND

- ③ The subquery of a view always begins with the `SELECT` keyword
- ③ The subquery in a view is a limited form of a `SELECT` command that can use `FROM`, `WHERE`, `GROUP BY` and `HAVING`; however the `ORDER BY` cannot be used
- ③ The subquery of a view determines the result set that can be used just like a table in other SQL commands

READ ONLY VIEWS

- ⦿ The main (i.e., first) FROM clause specifies multiple tables and/or views or specifies another read-only view
- ⦿ The first SELECT (following the AS keyword) specifies the DISTINCT keyword or a column function, such as Max(Discount)
- ⦿ A nested subquery command specifies the same table as the outer subquery
- ⦿ The outer subquery command contains a GROUP BY or HAVING clause
- ⦿ The first SELECT does not contain at least one column that is derived directly (i.e., without an expression) from a column of the underlying table
- ⦿ Read only view cannot be used in INSERT, UPDATE or DELETE

WITH CHECK AND WITH LOCAL CHECK OPTION

- ⦿ These options restrict row insert and update operations through an updatable view
- ⦿ Before support for check constraints, views that had `WITH CHECK OPTION` provided one means to define data integrity rules
- ⦿ Check constraints are a much better approach, because they are enforced whether the table is updated directly or through any view

VIEW EXAMPLE

```
Create View CustShort
    ( CsStrt, CsCity, CsSt, CsZip,
      CsCrRt, CsAttn, CsCnry, CustID )
AS Select ShipLine2,
          ShipCity,
          ShipState,
          ShipPsCd1,
          ShipPsCd2,
          ShipLine1,
          ShipCntry,
          CustID
      From Customer
```

**Reorder
and
rename
columns**

RETRIEVING ROWS FROM A VIEW WITH A SELECT

The last
Select is
equivalent
to the view
and Select
from the
view

```
Create View CustSt1
  AS Select *
      From Customer
      Where ShipCity = 'Toronto';
```

```
Select *
  From CustSt1
  Where Discount > 0;
```

```
Select *
  From Customer
  Where ShipCity = 'Toronto'
  And Discount > 0;
```

INDEXES

- ③ When users perform a query on a database, they generally are searching for one or more rows that satisfy a search criteria
- ③ If every row of the database table has to be examined to determine which rows satisfy the query, that would be too time-consuming
- ③ Indexes are created over tables that allow the DBMS to accelerate the searching process

WHAT IS AN INDEX?

- ① An index in a database is similar to an index in a book
- ① The index at the end of a book lets the readers find a specific location within the book without searching page by page
- ① The same principle applies to a database. Without an index, the DBMS reads through the entire table to locate the desired data
- ① When an index is built over the table, the DBMS can first go through the index to find out where the required data is located and then go to those locations directly to retrieve the data



USE OF INDEXES

- ⊙ Select a subset of the rows in a table (e.g., only customers in Toronto)
- ⊙ Combine the rows from multiple tables (e.g., combine sales rows from a table for the year 2008 with rows from a table for the year 2009)
- ⊙ Select a subset of the columns in a table (e.g., only the name and status of customers)
- ⊙ Combine (“join”) related rows in two or more tables (e.g., combine detailed customer data with each of the sales rows for the customer)
- ⊙ Provide an index so rows can be efficiently retrieved in a particular order (e.g., by customer name)

ACCESSING DATA USING AN INDEX

- ① Without an index, high level languages (HLL) programs receive the rows of a table in the same sequence in which they are actually physically stored in the table, thus first-in, first-out
- ① An index provides an access path in which the order of rows is based on ascending or descending values in one or more keys that are specified when a table or index is created
- ① A table defined with a primary key or index uses an access path in which rows are stored in sequence according to a key

STRUCTURE OF A KEYED SEQUENCE TABLE

Table containing data



Data File			
Employee Number	Store No	First Name	
827392161	7315	Maggie	...
228725876	5003	Brenda	...
132135478	1133	Janice	...
864955834	2257	Laura	...
103429376	4464	Sang Yong	...
314792638	1133	Isabel	...
223649622	5003	Tom	...
123728964	1133	Susan	...
832476894	2257	Stacey	...
235235658	4464	Karl	...

Key field used to identify each row in the employee pay table

Index keyed on employee number



Index File	
Employee Number	Pointer
103429376	05
123728964	08
132135478	03
223649622	07
235235658	10
228725876	02
314792638	06
827392161	01
832476894	09
864955834	04

Physical disk address of the associated row in the data portion of the employee pay table

←1st row retrieved from the index.
The value (05) indicates that this row is physically located as the fifth row in the data file.

CREATING AN INDEX

- ⊙ Indexes are used for efficient row selection and ordering
- ⊙ The DBMS automatically selects which index to be used when an SQL statement is executed
- ⊙ When a primary, unique, or foreign key constraint is specified, an index is created
- ⊙ Additional indexes can be created using the SQL `CREATE INDEX` statement

CREATE INDEX SYNTAX

```
Create Index index-name  
  On table-name  
    (column1, column2, column3);
```

SAMPLE CREATE INDEX STATEMENT

```
Create Index  
AppDta.CustCtyX01  
  On Customer  
  ( ShipCity,  
    CrdLimit Desc );
```

DROP INDEX SAMPLE

```
Drop Index  
AppDta.CustCtyX01;
```

CHAPTER SUMMARY

- ⦿ A view is an SQL object defined over one or more tables or other views that provides an alternative way to access the data in the underlying tables.
- ⦿ Views may be used to:
 - ⦿ Sequence data in a table in a different order.
 - ⦿ Select or omit specific rows from tables.
 - ⦿ Select or omit specific columns within rows from tables.
 - ⦿ Access data from two or more tables.
- ⦿ A subquery command is a limited form of Select command that plays a central part in view definition.

CHAPTER SUMMARY - VIEW STRUCTURE

The
brackets, or
[], indicate
clauses
that are
optional

```
Create View view-name As
  Select      select-list
    From      table-list
  [ Where     search-condition      ]
  [ Group By  grouping-column-list ]
  [ Having    search-condition      ];
```

CHAPTER SUMMARY - VIEWS

- ① The full power of the SQL subquery is available to define views. SQL view rows do not have any ordering.
- ① You can change table data through a view only if the view is updatable (i.e., not read-only).
- ① The With Check Option and With Local Check Option clauses prevent "phantom updates," in which a row is inserted or updated through a view but cannot subsequently be retrieved through the view.
- ① When a SELECT command is executed on a view, the conditions specified on the view's Where clause and the conditions specified on the Select command are combined.

CHAPTER SUMMARY - INDEXES

- ③ The Create Index statement creates an SQL index which is an object with a keyed access path.
- ③ An SQL index is created over a single table and specifies one or more key columns.
- ③ The Drop Index statement deletes an index.