Index Blocking Factors, Views

Rose-Hulman Institute of Technology

Curt Clifton

Index Redux

- Heap storage
- Clustered (primary) index
- Non-clustered (secondary) index
  - On heap stored table
  - On clustered table

Index Calculations

- To understand index, helpful to calculate sizes
- Terms:
  - Blocking factor: How many records fit on a page (a.k.a., a block) – *see Q1a*
  - Index block factor: How many index entries fit on a page – *see Q1b*

Index Calculations (cont.)

- Single level index…
  - If primary (clustered), then one entry for each block in file
  - If secondary, then one entry for each entry in file
  - *See Q1c*
- Multi-level index
  - One entry for each block at the next lower level
  - *See Q1d, e*

Views

Employees

EmployeeID	LastName	Firstname	Title
1	Davolio	Nancy	~~~
2	Fuller	Andrew	~~~
3	Leverling	Janet	~~~

```
USE Northwind
GO
CREATE VIEW dbo.EmployeeView
AS
SELECT LastName, Firstname
FROM Employees
```

EmployeeView

Lastname	Firstname
Davolio	Nancy
Fuller	Andrew
Leverling	Janet

User's View

Advantages of Views

- Focus the Data for Users
    - Focus on important or appropriate data only
    - Limit access to sensitive data (hide SSN from professors)

- Mask Database Complexity
    - Hide complex database design
    - Simplify complex queries, including distributed queries to heterogeneous data by embedding them in views

- Simplify Management of User Permissions
    - Different user access DB from different views

Creating Views

☐ Creating a View

```
CREATE VIEW dbo.OrderSubtotalsView (OrderID, Subtotal)
AS
SELECT OD.OrderID,
 SUM(CONVERT(money,(OD.UnitPrice*Quantity*(1-Discount)/100))*100)
FROM [Order Details] OD
GROUP BY OD.OrderID
GO
```

☐ Restrictions on View Definitions

■ Cannot include ORDER BY clause

Example: View of Joined Tables

Orders

OrderID	CustomerID	RequiredDate	ShippedDate
10663	BONAP	1997-09-24	1997-10-03
10827	BONAP	1998-01-26	1998-02-06
10427	PICCO	1997-02-24	1997-03-03
10451	QUICK	1997-03-05	1997-03-12
10515	QUICK	1997-05-07	1997-05-23

Customers

CustomerID	CompanyName	ContactName
BONAP	Bon app'	Laurence Lebihan
PICCO	Piccolo und mehr	Georg Pipps
QUICK	QUICK-Stop	Horst Kloss

```
USE Northwind
GO
CREATE VIEW dbo.ShipStatusView
AS
SELECT OrderID, ShippedDate, ContactName
FROM Customers C INNER JOIN Orders O
   ON C.CustomerID = O.CustomerID
WHERE RequiredDate < ShippedDate
```

ShipStatusView

OrderID	ShippedDate	ContactName
10264	1996-08-23	Laurence Lebihan
10271	1996-08-30	Georg Pipps
10280	1996-09-12	Horst Kloss

Altering and Dropping Views

- Altering Views

```
USE Northwind
GO
ALTER VIEW dbo.EmployeeView
AS
SELECT LastName, FirstName, Extension
FROM Employees
```

   - Retains assigned permissions

   - Causes new SELECT statement and options to replace existing definition

- Dropping Views

```
DROP VIEW dbo.ShipStatusView
```

Locating View Dependencies

- Use: **sp_depends** *viewname*
- Will list:
  - Objects upon which view depends
    - The "underlying" or "base" relations
  - Objects that depend on the view

Modifying Data Through Views

- Update or delete allowed on view **when it can be mapped to just one underlying table**

- Cannot modify computed columns

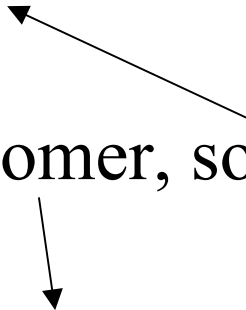- Queries executed by translation to underlying table (typically)

Quiz Question 3

- Pertinent SodaBases relations:
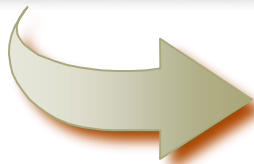 - Soda(name, manf)

  - Likes(customer, soda)

  - Customer(name, addr, phone)

Customers

Orders

Order Details

1	~		~		~	~
2	~		~		~	~
3	~		~		~	~
4	~		~		~	~
5	~		~		~	~
6	~		~		~	~

```
USE Northwind
GO
CREATE VIEW dbo.TopSalesView
AS
SELECT *
FROM dbo.TotalPurchaseView
WHERE Subtotal > 50000
GO
```

TotalPurchaseView

1	~	~	~	~
2	~	~	~	~
3	~	~	~	~
4	~	~	~	~
5	~	~	~	~
6	~	~	~	~

TopSalesView

	~	~	~	
	~	~	~	
	~	~	~	

```
SELECT *
FROM dbo.TopSalesView
WHERE CompanyName = 'Ernst Handel'
```

Moral of the Story

□ Don't create views on views