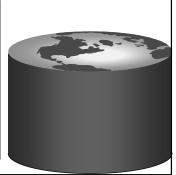
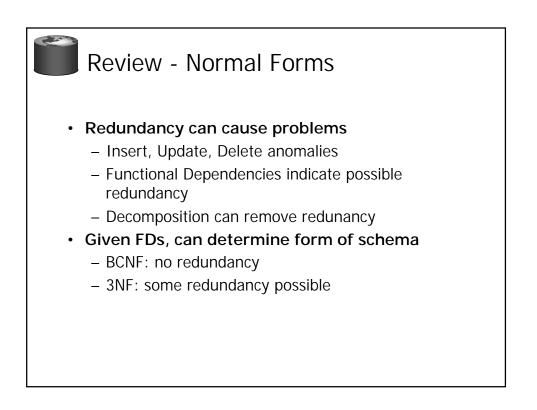
## Physical Database Design and Tuning

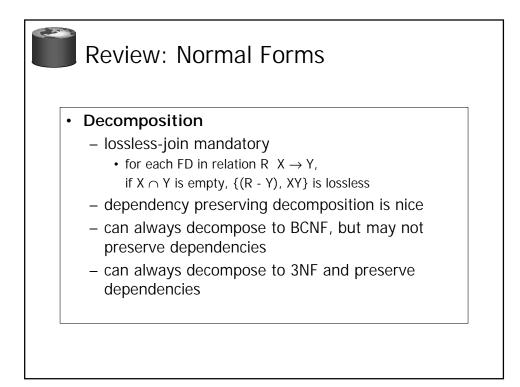
15-415, Spring 2003, Lecture 19 R & G Chapter 20

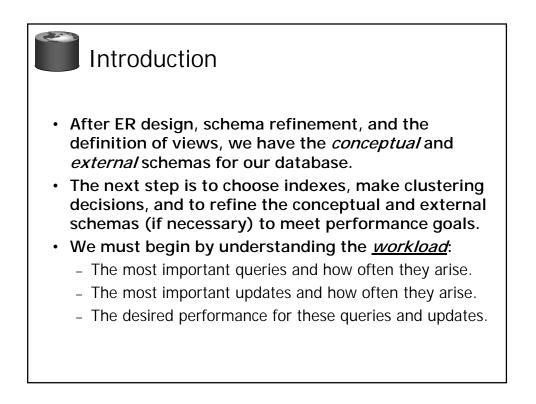
Although the whole of this life were said to be nothing but a dream and the physical world nothing but a phantasm, I should call this dream or phantasm real enough, if, using reason well, we were never deceived by it.

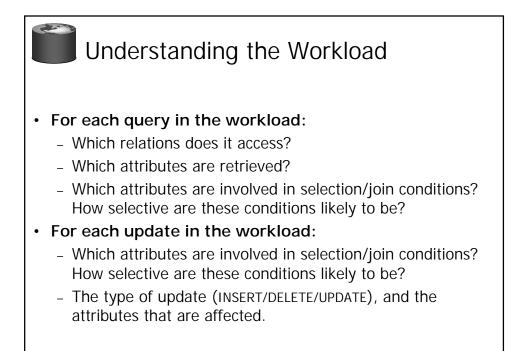


Baron Gottfried Wilhelm von Leibniz

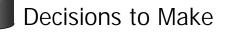






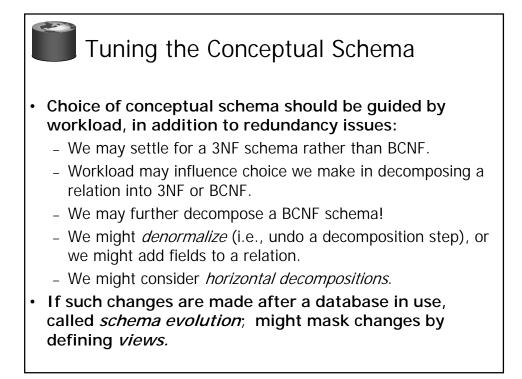


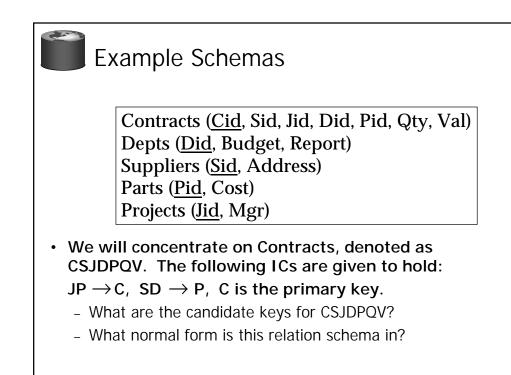
Croa	ting an		Char	ł	
	ung an	1300	Chai	ι	
Ins	sert, Selec	t Undata	Dolot	o Fronio	ncies
1115	ieri, selec	i, Opuuie	e, Delelo	e rrequei	icies
			Employ	ee Table	
Transaction	Frequency	% table	<i>Employ</i> Name	ee <i>Table</i> Salary	Address
<i>Transaction</i> Payroll Run	Frequency monthly	100 (% <i>table</i>	Name		Address S
			Name	Salary	
Payroll Run	monthly daily	100	<b>Name</b> S I	Salary	
Payroll Run Add Emps	monthly daily	100 0.1	Name S I D	Salary S	S I
Payroll Run Add Emps Delete Emps	monthly daily daily	100 0.1 0.1	Name S I D	Salary S I D	S I
Payroll Run Add Emps Delete Emps	monthly daily daily	100 0.1 0.1	Name S I D	Salary S I D	S I
Payroll Run Add Emps Delete Emps	monthly daily daily	100 0.1 0.1	Name S I D	Salary S I D	S I

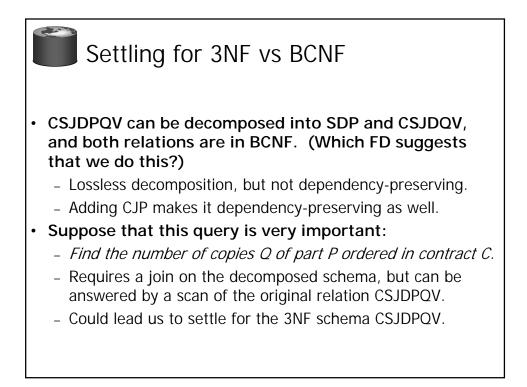


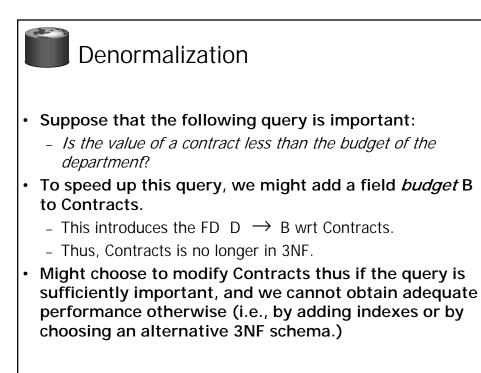
## · What indexes should we create?

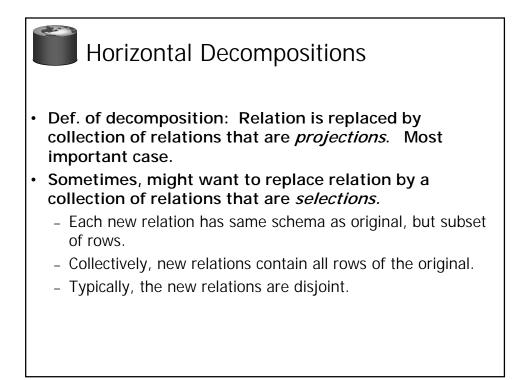
- Which relations should have indexes? What field(s) should be the search key? Should we build several indexes?
- For each index, what kind of an index should it be?
  - Clustered? Hash/tree? Dynamic/static? Dense/sparse?
- · Should we make changes to the conceptual schema?
  - Consider alternative normalized schemas? (Remember, there are many choices in decomposing into BCNF, etc.)
  - Should we ``undo" some decomposition steps and settle for a lower normal form? (*Denormalization*.)
  - Horizontal partitioning, replication, views ...

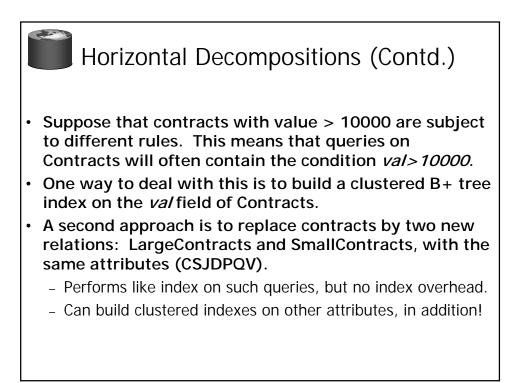




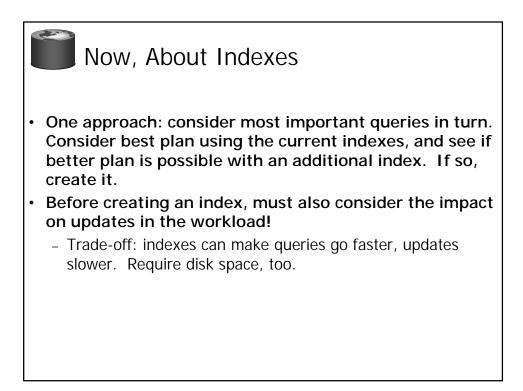


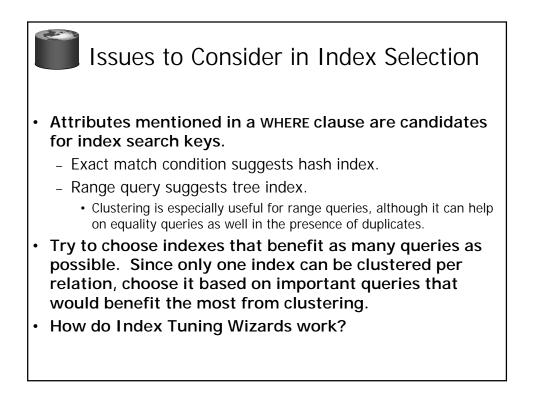






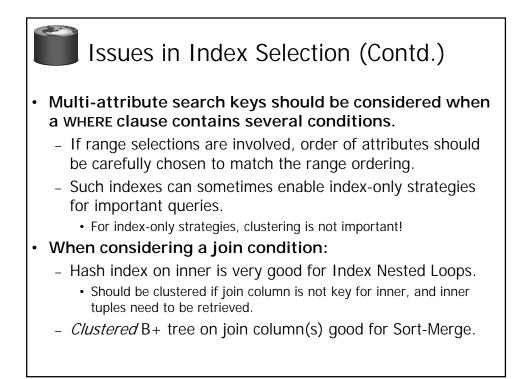
Masking Conceptual Schema Changes CREATE VIEW Contracts(cid, sid, jid, did, pid, qty, val)				
AS SELECT *				
FROM LargeContracts				
UNION				
SELECT *				
FROM SmallContracts				
<ul> <li>Suppose contracts w/ val &gt; 10000 have different rules. Then queries on Contracts will often contain predicate val &gt; 10000.</li> <li>Can use <u>Horizonal Decomposition</u> into LargeContracts and SmallContracts</li> </ul>				
<ul> <li>This decomposition can be masked by a view.</li> </ul>				
<ul> <li>But, queries with condition val&gt;10000 must be asked wrt LargeContracts for efficiency: so some users may have to be aware of change.</li> </ul>				

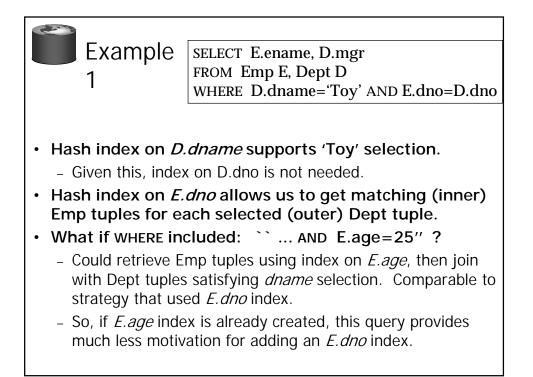


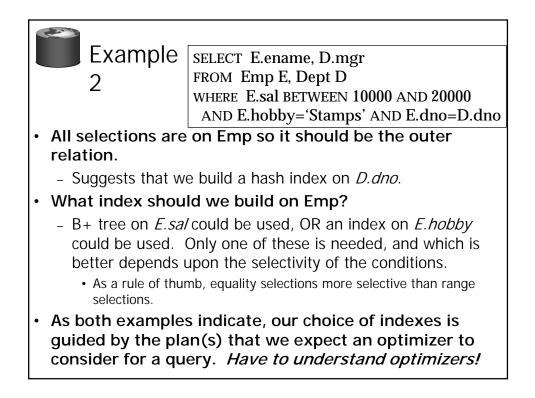


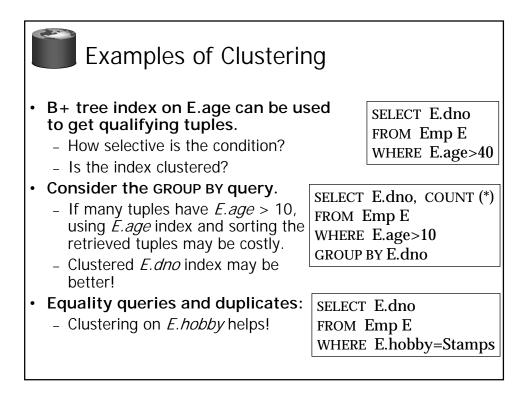


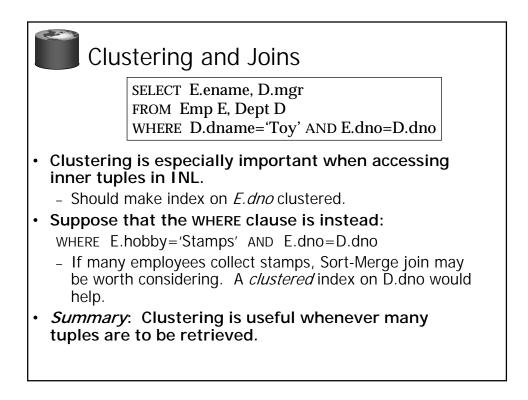
- Enumerate all *admissible indexes* (those that have indexable attributes as keys) with 1 attribute,
- Then add all 2-attribute key combinations, ...
- Then add all k-attribute key combinations (user-defined k)
- 2. Configuration Enumeration
  - Enumerate all configurations from above set
  - Simulate effect of indexes in candidate configuration using whatif queries to the optimizer
  - Suggest configuration that lowers workload cost the most
- Greedy algorithms can be improved (see book)

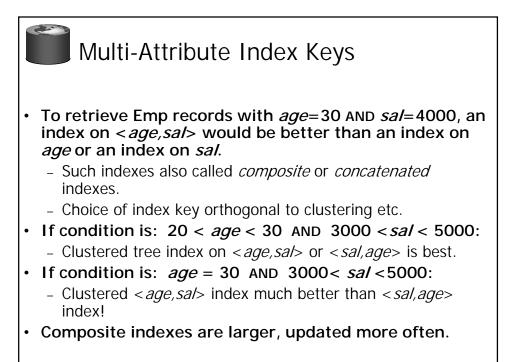




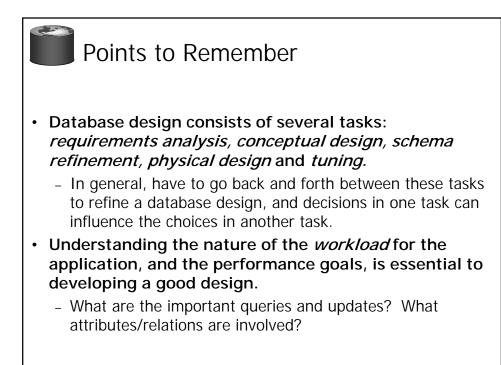


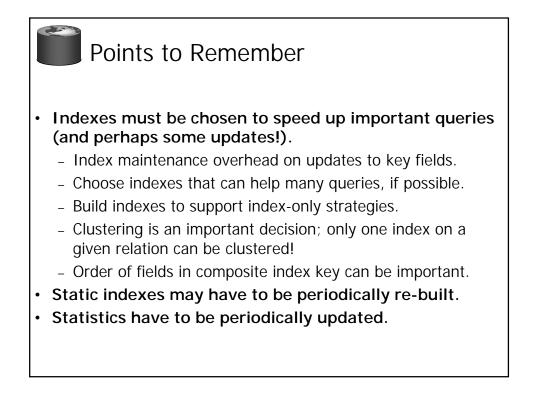


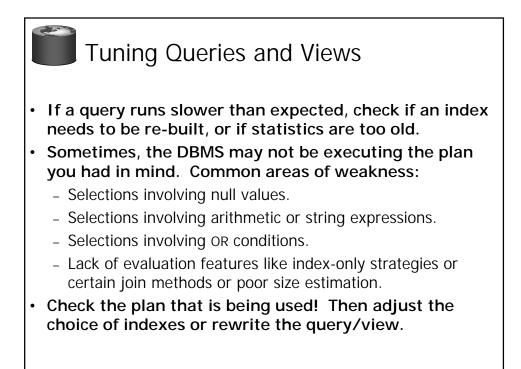


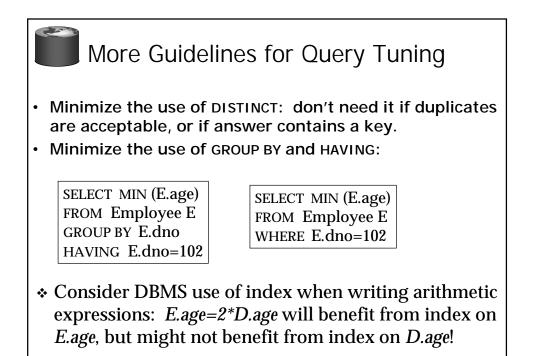


Index-Only Plan • A number of	S < <i>E.dno</i> >	SELECT D.mgr FROM Dept D, Emp E WHERE D.dno=E.dno	
queries can be answered <e.dno without Tree</e.dno 		SELECT D.mgr, E.eid FROM Dept D, Emp E WHERE D.dno=E.dno	
retrieving any tuples from one < <u>E.</u> or more of the relations	<b>dno&gt;</b>	SELECT E.dno, COUNT(*) FROM Emp E GROUP BY E.dno	
involved if a suitable index is <sup><e.dno,e.t< sup=""> available.</e.dno,e.t<></sup>	sal>   F	SELECT E.dno, MIN(E.sal) FROM Emp E GROUP BY E.dno	
<e. age,e.sal=""> or <e.sal, e.age=""> Tree!</e.sal,></e.>	FROM WHER	t avg(E.sal) Emp E e E.age=25 and between 3000 and 5000	







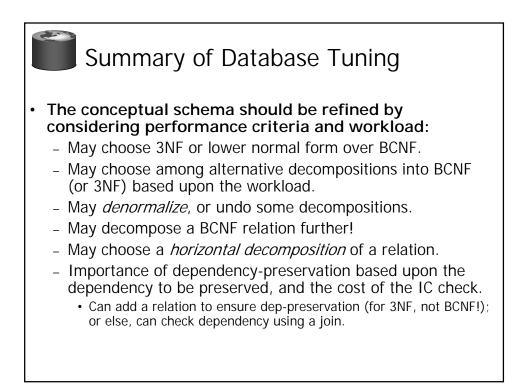


## Guidelines for Query Tuning (Contd.)

- Avoid using intermediate relations:
   SELECT E.dno, AVG(E.sal) FROM Emp E, Dept D
   SELECT E.dno, AVG(E.sal) and
- <u>vs.</u> WHERE E.dno=D.dno AND D.mgrname='Joe' GROUP BY E.dno

SELECT T.dno, AVG(T.sal) FROM Temp T GROUP BY T.dno

- \* Does not materialize the intermediate reln Temp.
- If there is a dense B+ tree index on <dno, sal>, an index-only plan can be used to avoid retrieving Emp tuples in the second query!



## Summary (Contd.)

- Over time, indexes have to be fine-tuned (dropped, created, re-built, ...) for performance.
  - Should determine the plan used by the system, and adjust the choice of indexes appropriately.
- System may still not find a good plan:
  - Only left-deep plans considered!
  - Null values, arithmetic conditions, string expressions, the use of ORs, etc. can confuse an optimizer.
- So, may have to rewrite the query/view:
  - Avoid nested queries, temporary relations, complex conditions, and operations like DISTINCT and GROUP BY.