Query Processing and Optimization

Project, Cartesian Product, Union, Intersect, Difference

Elmasri & Navathe
Implementing the Project Operation

• Fairly simple, two situations can arise:

  • 1) If the projection attributes includes the key of the relation.

  • 2) if the projection attributes do not include the key.
Implementing the Project Operation

1) If the projection attributes includes the key of the relation.

   – The resulting set will have the same number of tuples as the original relation.

   – The primary index can be used to extract the projected attributes.
Implementing the Project Operation

2) if the projection attributes do not include the key.

- In this case the resulting relation is typically sorted and duplicates are eliminated.

- Hashing can also be used to eliminate duplicates.
Implementing the Cartesian Product

\( R \times S \)

- Very expensive.

- If \( R \) has \( n \) records and \( j \) attributes, and \( S \) has \( m \) records and \( k \) attributes.

- \( R \times S \) has \( n \times m \) records and \( j+k \) attributes.

- Try to avoid cartesian product when doing query optimization.
Implementing the Union

R Union S

- R and S, must be union compatible.

- Sort R and S, then scan both files simultaneously, if duplicates are encountered, keep only the first instance. (Remove duplicates)
Implementing the Intersect

R Intersect S

– R and S, must be union compatible.

– Sort R and S, then scan both files simultaneously, if records match, put the matching record in the result set. Also remove duplicates.
Implementing the Difference

R - S

− R and S, must be union compatible.

− Sort R and S, then scan both files simultaneously, if records match, skip, if the records do not match, put the record from R in result set.
Hashing

Hashing can be used to implement Union, Intersect and Difference.

• To implement R U S:
  – First hash the records of R into an hash file.
  – Then hash the records of S into the same file, but do not insert the duplicate records.
Hashing can be used to implement Union, Intersect and Difference.

• To implement R Intersect S:
  – First hash the records of R into an hash file.
  – Then hash the records of S into the same file, if an identical record is found, insert it in the result set.
Hashing

Hashing can be used to implement Union, Intersect and Difference.

• To implement R - S:
  – First hash the records of R into an hash file.
  – Then hash the records of S into the same file, if an identical record is found, remove it from the result set.