**Project(R,A)**
Create a new relation that retains only the attributes A taken from R.
Category: 
Notes: 

**Select(R,C)**
Create a new relation including only tuples from R that satisfy C
Category: 
Notes: 

**DupElim(R)**
Create a new relation from R by including each unique tuple exactly once
Category: 
Notes: 

**GroupMin(R,A,B)**
Create a new relation consisting of unique tuples of the attributes A and the minima of the attributes B over the corresponding grouped sets of tuples.
Category: 
Notes: 

**GroupMax(R,A,B)**
Create a new relation consisting of unique tuples of the attributes A and the maxima of the attributes B over the corresponding grouped sets of tuples.
Category: 
Notes: 

**GroupCount(R,A)**
Create a new relation consisting of unique tuples of the attributes A and counts of the sizes of corresponding grouped sets of tuples.
Category: 
Notes: 

**GroupSum(R,A,B)**
Create a new relation consisting of unique tuples of the attributes A and the sums of the attributes B over the corresponding grouped sets of tuples.
Category: 
Notes: 

**GroupAvg(R,A,B)**
Create a new relation consisting of unique tuples of the attributes A and the averages of the attributes B over the corresponding grouped sets of tuples.
Category: 
Notes: 

**SetUnion(R,S)**
Create a new relation containing each of the unique tuples found in either R or S.
Category: 
Notes: 

**BagUnion(R,S)**
Create a new relation containing each of the tuples found in either R or S (including duplicates).
Category: 
Notes:
**SetIntersection(R,S)**
Create a new relation containing each of the unique tuples found in both R and S.
Category:
Notes:

**BagIntersection(R,S)**
Create a new relation containing each tuple found in both R and S, repeated the lesser of their number of occurrences in each.
Category:
Notes:

**SetDifference(R,S)**
Create a new relation containing each unique tuple found in R but not in S
Category:
Notes:

**BagDifference(R,S)**
Create a new relation containing each unique tuple found in R more often than S, as many times as there are excess appearances in R
Category:
Notes:

**Product(R,S)**
Create a new relation containing every possible concatenation of a tuple from R with a tuple from S.
Category:
Notes:

**NaturalJoin(R,S)**
Create a new relation containing concatenations of a tuple from R with a tuple from S, where the tuples match on shared attributes.
Category:
Notes:

**NestedLoopJoin(R,S)**
Create a new relation containing concatenations of a tuple from R with a tuple from S, where the tuples match on shared attributes.
Category:
Notes:

**Sort (R)**
Applies a two-phase multiway merge sort on R.
Category:
Notes:

**SortDupElim(R)**
Uses merge sort to eliminate duplicates in large relation R
Category:
Notes:

**SortGroupAgg(R,A,G)**
Uses merge sort to compute some aggregated property G of tuples from large relation R, as grouped by attributes A
Category:
Notes:
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Category</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>SortUnion(R,S)</td>
<td>Uses merge sort to take the union of large relations R and S</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SortIntersection(R,S)</td>
<td>Uses merge sort to take the intersection of large relations R and S</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SortDifference(R,S)</td>
<td>Uses merge sort to take the set difference of large relations R and S</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SortJoin(R,S)</td>
<td>Uses merge sort to produce a join of large relations R and S</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HashDupElim(R)</td>
<td>Uses hashing to eliminate duplicates in large relation R</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HashGroupAgg(R,A,G)</td>
<td>Uses hashing to compute some aggregated property G of tuples from large relation R, as grouped by attributes A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HashUnion(R,S)</td>
<td>Uses hashing to take the union of large relations R and S</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HashIntersection(R,S)</td>
<td>Uses hashing to take the intersection of large relations R and S</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HashDifference(R,S)</td>
<td>Uses hashing to take the set difference of large relations R and S</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HashJoin(R,S)</td>
<td>Uses hashing to produce a join of large relations R and S</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**IndexSelect(R,A)**

Uses an index to select tuples from R matching condition C on A

Category:

Notes:

**SortedIndexJoin(R,S)**

Uses a sorted index to produce a join of large relations R and S

Category:

Notes: