Execution Plans

- an *execution plan* consists of a query tree and an algorithm for each relational algebra operation in the query tree

Implementing SELECT

Some possibilities for simple selection (a single condition):

name	strategy	restrictions
SL	brute force linear search (scan entire file)	always applicable
SB	binary search on the file	must have file ordered on attribute
SH	use hash key	must have file hashed on attribute must be equality condition
SP	use primary index	must have primary index on attribute
SC	use clustering index	must have clustering index on attribute
SS	use secondary index	must have secondary index on attribute

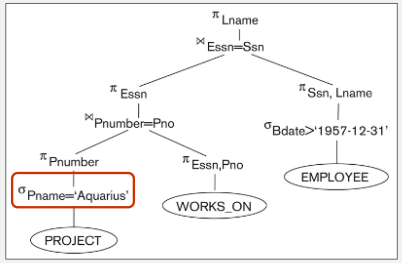
- only SL can be pipelined
  - SL can be carried out incrementally
  - everything else requires random access to the file or an index

All questions make use of the following relational schema and query tree. Assume that the file for each table is ordered by the primary key, and that there are primary indexes for each file as well as indexes on PROJECT.Pname, WORKS_ON.Pno, and EMPLOYEE.Lname.

```
EMPLOYEE(Fname,Minit,Lname,Ssn,Bdate,Address,Sex,Salary,Super_ssn,Dno)
PROJECT(Pname,Pnumber,Plocation,Dnum)
WORKS_ON(Essn,Pno,Hours)
```

PROJECT is ordered by Pnumber
– SB, SH not applicable for Pname

the index on Pname is a secondary index
– SP, SC not applicable



What algorithms are suitable for the $\sigma_{Pname='Aquarius'}$ operation? Choose all that apply.

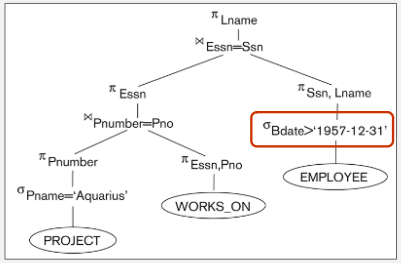
SL	4 respondents	80 % ✓
SB	2 respondents	40 %
SH	2 respondents	40 %
SP		0 %
SC	2 respondents	40 %
SS	4 respondents	80 % ✓
none of the above		0 %

All questions make use of the following relational schema and query tree. Assume that the file for each table is ordered by the primary key, and that there are primary indexes for each file as well as indexes on PROJECT.Pname, WORKS_ON.Pno, and EMPLOYEE.Lname.

```
EMPLOYEE(Fname,Minit,Lname,Ssn,Bdate,Address,Sex,Salary,Super_ssn,Dno)
PROJECT(Pname,Pnumber,Plocation,Dnum)
WORKS_ON(Essn,Pno,Hours)
```

EMPLOYEE is ordered by Ssn
– SB, SH not applicable for Bdate

there is no index on Bdate (only Ssn and Lname)
– SP, SC, SS not applicable



What algorithms are suitable for the $\sigma_{Bdate>'1957-12-31'}$ operation? Choose all that apply.

SL	5 respondents	100 % ✓
SB	1 respondent	20 %
SH		0 %
SP	2 respondents	40 %
SC	2 respondents	40 %
SS	1 respondent	20 %
none of the above		0 %

Implementing JOIN $R \bowtie S$

Some possibilities for join:

name	strategy	restrictions
JNL	*nested-loop join* (brute force) for each record in R, go through every record in S and test if the join condition is satisfied	always applicable
JSL	*single loop join* for each record in R, use index to retrieve all matching records of S	requires index on join attribute for one file (S)
JSM	*sort-merge join* if necessary, sort files by join attribute scan files in order, matching records according to join attribute	always applicable
	scan indexes, matching records according to join attribute	requires indexes on join attribute for both files

- reading of R can be pipelined for JNL, JSL
- JSM can be pipelined for R and S if the records come out of the previous step in order (so no need to sort)

Implementing PROJECT

Without duplicate elimination, simply extract the desired columns for each record.

- can be pipelined

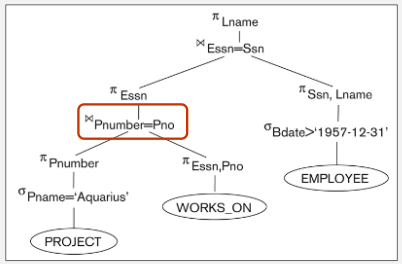
For duplicate elimination:

name	strategy
PS	sort file (duplicates will be adjacent)
PH	hash file (check each record against others in the bucket it hashes to)

  - PS can be pipelined if the records come out of the previous step in order

All questions make use of the following relational schema and query tree. Assume that the file for each table is ordered by the primary key, and that there are primary indexes for each file as well as indexes on PROJECT.Pname, WORKS_ON.Pno, and EMPLOYEE.Lname.

```
EMPLOYEE(Fname,Minit,Lname,Ssn,Bdate,Address,Sex,Salary,Super_ssn,Dno)
PROJECT(Pname,Pnumber,Plocation,Dnum)
WORKS_ON(Essn,Pno,Hours)
```



JNL is always applicable

JSM is always applicable, though may require sorting
– PROJECT is ordered by Pnumber (no sorting)
– WORKS_ON is ordered by Essn,Pno (need sorting, or retrieve rows via WORKS_ON.Pno index)

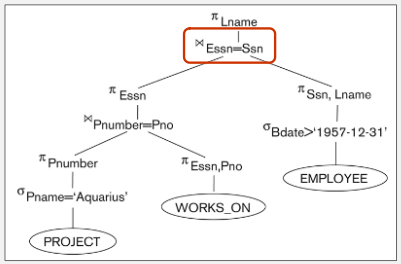
JSL requires an index on S – there is an index on WORKS_ON.Pno and π (without duplicate removal) can be pipelined

What algorithms are suitable for the $\bowtie_{Pnumber=Pno}$ operation? Choose all that apply.

JNL	4 respondents	80 % ✓
JSL	4 respondents	80 % ✓
JSM	4 respondents	80 % ✓
none of the above		0 %

All questions make use of the following relational schema and query tree. Assume that the file for each table is ordered by the primary key, and that there are primary indexes for each file as well as indexes on PROJECT.Pname, WORKS_ON.Pno, and EMPLOYEE.Lname.

```
EMPLOYEE(Fname,Minit,Lname,Ssn,Bdate,Address,Sex,Salary,Super_ssn,Dno)
PROJECT(Pname,Pnumber,Plocation,Dnum)
WORKS_ON(Essn,Pno,Hours)
```



JNL is always applicable

JSM is always applicable, though may require sorting
– PROJECT is ordered by Pnumber so rows will likely come out of $\bowtie_{Pnumber=Pno}$ ordered by Pnumber (sorting needed)
– EMPLOYEE is ordered by Ssn (need sorting)

JSL requires an index on S – there is an index on EMPLOYEE.Ssn but if that is used for the join, SL is required for $\sigma_{Bdate>'1957\text{-}12\text{-}31'}$ and both it and $\pi_{Ssn,Lname}$ must be pipelined (without duplicate removal)

What algorithms are suitable for the $\bowtie_{Essn=Ssn}$ operation? Choose all that apply.

JNL	4 respondents	80 % ✓
JSL	4 respondents	80 %
JSM	3 respondents	60 % ✓
none of the above		0 %