CS W186 - Spring 2024
Exam Prep Section 6
Iterators and Joins

Joins 1

We will be joining two tables: a table of students, and a table of assignment submissions; and we will be joining by the student ID:

```
CREATE TABLE Students (
      student_id INTEGER PRIMARY KEY,
      ...
);

CREATE TABLE AssignmentSubmissions(
      assignment_number INTEGER,
      student_id INTEGER REFERENCES Students(student_id),
      ...
);
```

For the questions below, consider the following query:

```
SELECT *
FROM Students, AssignmentSubmissions
WHERE Students.student_id = AssignmentSubmissions.student_id;
```

We also have:

- Students has **[*S*] = 20 pages**, with p_S **= 200 records per page**
- AssignmentSubmissions has **[*A*] = 40 pages**, with p_A **= 250 records per page**

Questions:

1. What is the I/O cost of a simple nested loop join for Students on AssignmentSubmissions?

2. What is the I/O cost of a simple nested loop join for AssignmentSubmissions on Students?

3. What is the I/O cost of a block nested loop join for Students on AssignmentSubmissions?
   Assume our buffer size is B = 12 pages.

4. What about block nested loop join for AssignmentSubmissions on Students?

   Assume our buffer size is B = 12 pages.

5. What is the I/O cost of an Index-Nested Loop Join for Students on AssignmentSubmissions?

   Assume we have a clustered alternative 2 index on AssignmentSubmissions.student id, in the form of a height 2 B+ tree. Assume that index node and leaf pages are not cached; all hits are on the same leaf page; and all hits are also on the same data page.

6. Now assume we have a unclustered alternative 2 index on AssignmentSubmissions.student id, in the form of a height 2 B+ tree. Assume that index node pages and leaf pages are never cached, and we only need to read the relevant leaf page once for each record of Students, and all hits are on the same leaf page.

   What is the I/O cost of an Index-Nested Loop Join for Students on AssignmentSubmissions?

   HINT: The foreign key in AssignmentSubmissions may play a role in how many accesses we do per record.

7. What is the cost of an unoptimized sort-merge join for Students on AssignmentSubmissions?

   Assume we have B = 12 buffer pages.

8. What is the cost of an optimized sort-merge join for Students on AssignmentSubmissions?

   Assume we have B = 12 buffer pages.

9. In the previous question, we had a buffer of B = 12 pages. If we shrank B enough, the answer we got might change.

   How small can the buffer B be without changing the I/O cost answer we got?

10. What is the I/O cost of Grace Hash Join on these tables?

    Assume we have a buffer of B = 6 pages.

Joins 2

Consider a modified version of the baseball database that only stores information from the last 40 years.

```
CREATE TABLE Teams (
      team_id INTEGER PRIMARY KEY,
      team_name VARCHAR(20),
      year INTEGER,
       ...
);

CREATE TABLE Players(
      player_id INTEGER PRIMARY KEY,
      team_id INTEGER REFERENCES Teams(team_id),
      year INTEGER,
       ...
);
```

Each record in Teams represents a single team for a single year. Each record in Players represents a single player during a single year. We also have:

- Teams has **$[T]$ = 30 pages**, with **p_T = 40 records per page**

- Players has **[P] = 300 pages**, with **p_P = 50 records per page**

Questions:

1. What is the I/O cost of a simple nested loop join for joining Teams on Players on Teams.team id = Players.team id?

2. What is the I/O cost of a page nested loop join on the same query?

3. What is the I/O cost of a block nested loop join on the same query? Assume our buffer size is B = 10 pages.

4. Assume we have an unclustered index of height 1 on Teams.team id. What is the I/O cost of an index nested loop join on Teams.team id = Players.team id using this index? You can assume that every player only plays on one team each year.

5. Now, assume we have a clustered index of height 2 on Players.player id and an clustered index of height 3 on Players.team id. If each team has 25 players each year, what is the lowest I/O cost of the join on on Teams.team id = Players.team id using one of these indexes?

6. Assume the index on Players.team id is actually unclustered. What is cost of the join on on Teams.team id = Players.team id using this index?

7. Consider a universe where there are no limits on team size and players get to time travel to play for whatever team they want, and 75% of players choose to travel to 2020 to play for the best baseball team, the San Diego Padres. In other words, imagine that 75% of the records in the Players table

have the same team id. What are the effects on the performance of the different join algorithms? *Hint: Consider which of the joins are affected by duplicates. Remember that only one of the tables has duplicates, while the other does not.*

Joins 3

In the following problems, we will be joining two tables: Students and AssignmentSubmissions on the key 'student id'. However, we are dealing with a set of system constraints. Given a set of potential join algorithms from SNLJ, BNLJ, PNLJ, Hash Join, GHJ, SMJ, select the best option(s).

```
CREATE TABLE Students (
    student_id INTEGER PRIMARY KEY,
    ...
);

CREATE TABLE AssignmentSubmissions(
    assignment_number INTEGER,
    student_id INTEGER REFERENCES Students(student_id),
    ...
);
```

For the questions below, consider the following query:

```
SELECT *
FROM Students, AssignmentSubmissions
WHERE Students.student_id = AssignmentSubmissions.student_id;
```

We also have:

- Students has **[*S*] = 600 pages**, with p_S **= 60 records per page**
- AssignmentSubmissions has **[*A*] = 600 pages**, with p_A **= 60 records per page**
- **B = 10**

1. Our program memory is extremely limited (not buffer memory)! As a result, we only have enough memory to store 1 hash function. What join algorithms work here provided it fits our system con straints)? Why?

2. Now, we have very little buffer memory (only 3 pages). What join algorithms work here? How are the different join algorithms affected by the given constraint? Why?

3. Now, assume that Students is only 1 page. What is the best join algorithm IO wise?

4. Now, assume that all student ids in both tables are exactly the same (i.e. assume the primary key constraint does not hold). What join algorithms work here? How are the different join algorithms affected by the given constraint? Why?