

CIS560

Joins – Part 1



Topics

- CROSS JOIN
- INNER JOIN
- Variations
 - Self-Join
 - Composite Join
 - Non-equi Join
 - Multi-Join Queries



Review

- ANSI Processing Order (Logical)

```

5 SELECT [DISTINCT | TOP]...
1 FROM ...
2 WHERE ...
3 GROUP BY ...
4 HAVING ...
7 ORDER BY ...
  OFFSET... FETCH ...

```

- OFFSET-FETCH is part of the ORDER BY clause



Relationship Types

- One-to-Many

- Zero or more \rightarrow One and only one
- One or more \rightarrow One and only one (logical only)
- Zero or more \rightarrow Zero or one

- One-to-One

- Zero or one \rightarrow One and only one

- Many-to-Many

Implemented with a “linking” or “bridge” table



Table Operators

- Used in the FROM element.
- Perform operations on input tables and produce an output table.
- SQL Server supports four table operators:
 - JOIN
 - APPLY
 - PIVOT
 - UNPIVOT
- JOIN is the only standard operator.



Joins

- Joins operate on two input tables.
- Produce a single output table.
- There are three types of joins:
 - CROSS JOIN
 - INNER JOIN
 - OUTER JOIN
- All differ in their logical processing phases.



CROSS JOIN

- A single logical processing phase:
 1. Produces a Cartesian Product
- Two syntaxes are supported:
 - ANSI SQL-89
 - ANSI SQL-92



Cross Join

ID	Name
1	Jim
2	Kim
3	Alice



ID	Food
2	Pickles
3	Fish
7	Ice Cream



A.ID	Name	B.ID	Food
1	Jim	2	Pickles
1	Jim	3	Fish
1	Jim	7	Ice Cream
2	Kim	3	Fish
2	Kim	2	Pickles
2	Kim	7	Ice Cream
3	Alice	7	Ice Cream
3	Alice	2	Pickles
3	Alice	3	Fish

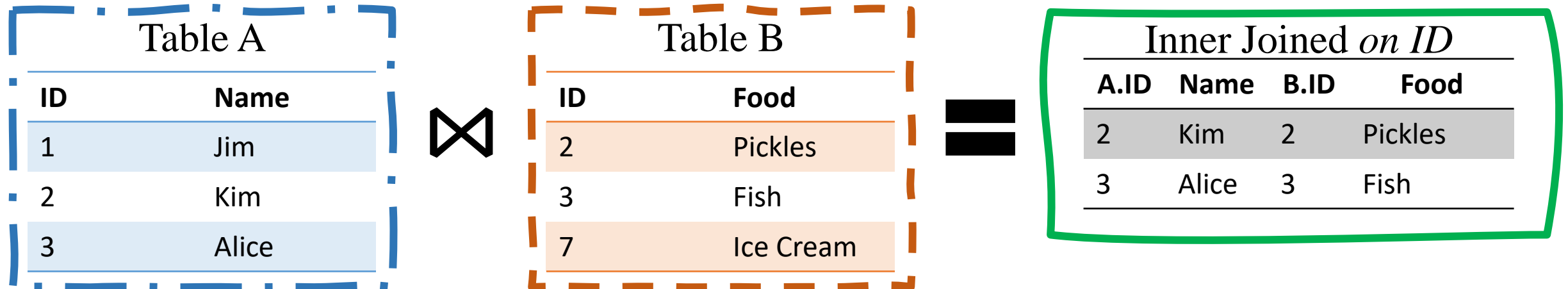


INNER JOIN

- Two logical processing phases:
 1. Produces a Cartesian Product
 2. Filters rows based on a provided predicate
- Two syntaxes are supported:
 - ANSI SQL-89
 - ANSI SQL-92



Inner Join (*on ID*)



Variations of Joins

- **Composite Joins**

Predicate involves more than one attribute from each table.

- **Self Join**

The two inputs are the same table.

- **Non-Equi Joins**

Predicate uses operator other than equality.



Multi-Join Queries

- Multiple joins are supported.
- Logically the joins are processed from left to right.
 1. The first join is processed, producing a table result.
 2. The result from the first join then serves as an input to the second join.
 3. And so forth...
- Can force order with parentheses.



Syntax

```
FROM <table_source>  
    CROSS JOIN <table_source>  
    | [ INNER ] JOIN <table_source> ON <condition>
```

