Microsoft SQL Server 2019
Design & Develop

Masoud Mirzakhani

Senior DW/ ETL/ BI Architect

Types of SQL Command

- DDL(Data Definition Language)
  - CREATE
  - DROP
  - ALTER
  - RENAME
  - TRUNCATE

Types of SQL Command

- DML(Data Manipulation Language)
  - INSERT
  - UPDATE
  - DELETE
  - MERGE
  - LOCK

Types of SQL Command

- DQL(Data Query Language)
  - SELECT

Types of SQL Command

- DCL(Data Control Language)
  - GRANT
  - REVOKE

Types of SQL Command

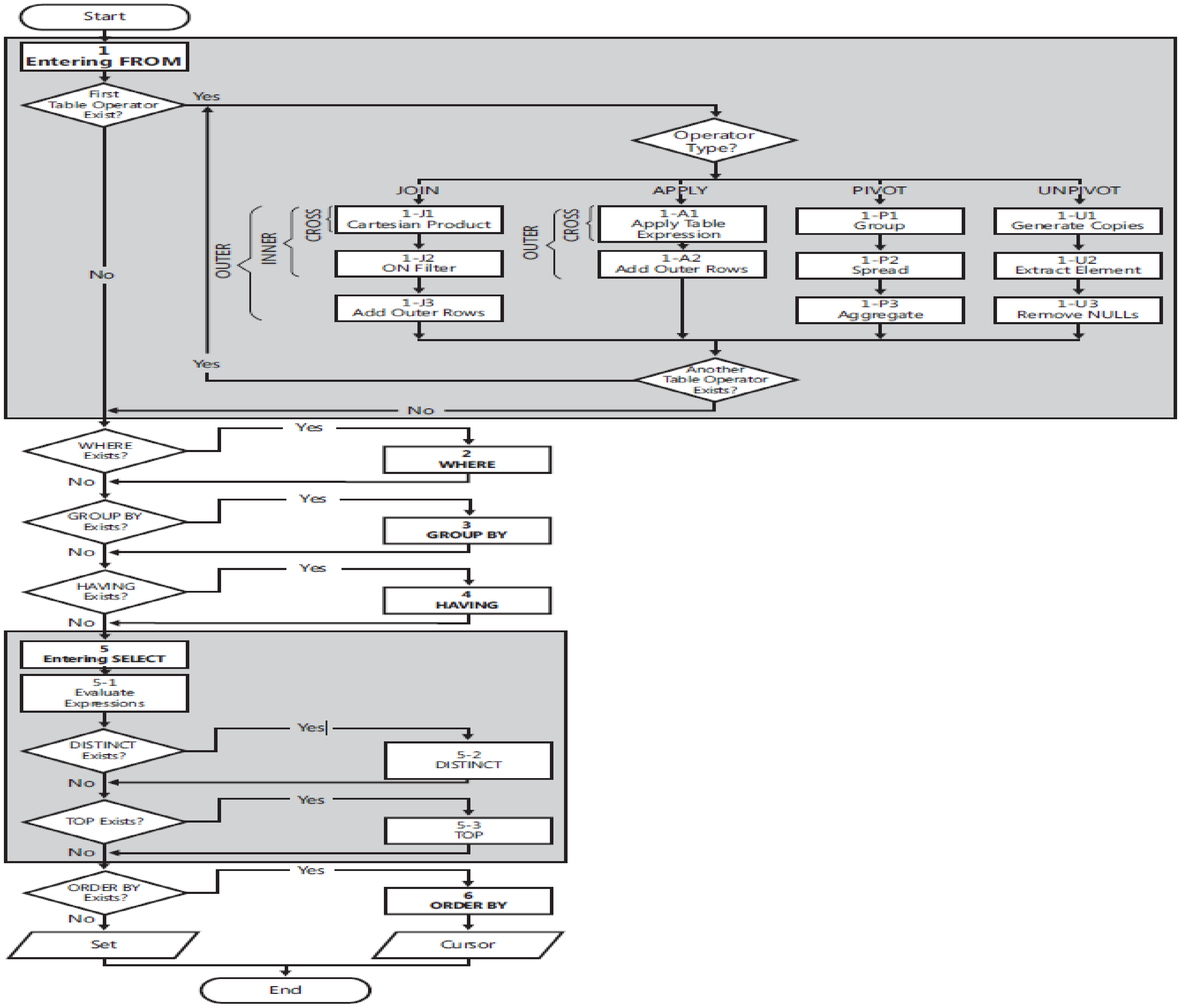
- TCL(Transaction Control Language)
  - BEGIN TRAN
  - COMMIT
  - ROLLBACK
  - SAVEPOINT

Types of Programming

- Procedural (Imperative)
  - HOW
- Declarative
  - WHAT

Logical Query Processing

```
(5) SELECT (5-2) DISTINCT (7) TOP(<top_specification>)
        (5-1) <select_list>
(1) FROM
        (1-J) <left_table> <join_type> JOIN <right_table> ON <on_predicate>
        (1-A) <left_table> <apply_type> APPLY <right_input_table> AS <alias>
        (1-P) <left_table> PIVOT(<pivot_specification>) AS <alias>
        (1-U) <left_table> UNPIVOT(<unpivot_specification>) AS <alias>
(2) WHERE
        <where_predicate>
(3) GROUP BY
        <group_by_specification>
(4) HAVING
        <having_predicate>
(6) ORDER BY
        <order_by_list>
(7) OFFSET
        <offset_specification> ROWS FETCH NEXT <fetch_specification> ROWS ONLY;
```



Logical Query Processing

(1) FROM

- Identify query’s source tables (sets)
- Process table (set) operators
    - JOIN
    - APLLY
    - PIVOT
    - UNPIVOT

(5-1) <select_list>

- Evaluate Expressions
  - Column
  - Fixed value
  - SQL functions
  - Combination of one or more
    - Columns,
    - Fixed values
    - SQL functions

(5-1) <select_list>

- * (Asterisk)
- Aliasing
    - expression AS <alias>
    - expression <alias>
    - <alias> = expression

(2) WHERE

- Filter the rows from previous step
  - Based on <where_predicate>
- Only rows which evaluated to TRUE go to next step

(1-J) JOIN

Employee		
ID	**Name**	**CityID**
1	Ali	3
2	Omid	4
3	Reza	5

```
{       (1, Ali, 3),
        (2, Omid, 4),
        (3, Reza, 5)
}
```

City	
ID	**Name**
3	Tehran
4	Shiraz
5	Tabriz

```
{       (3, Tehran),
        (4, Shiraz),
        (5, Tabriz)
}
```

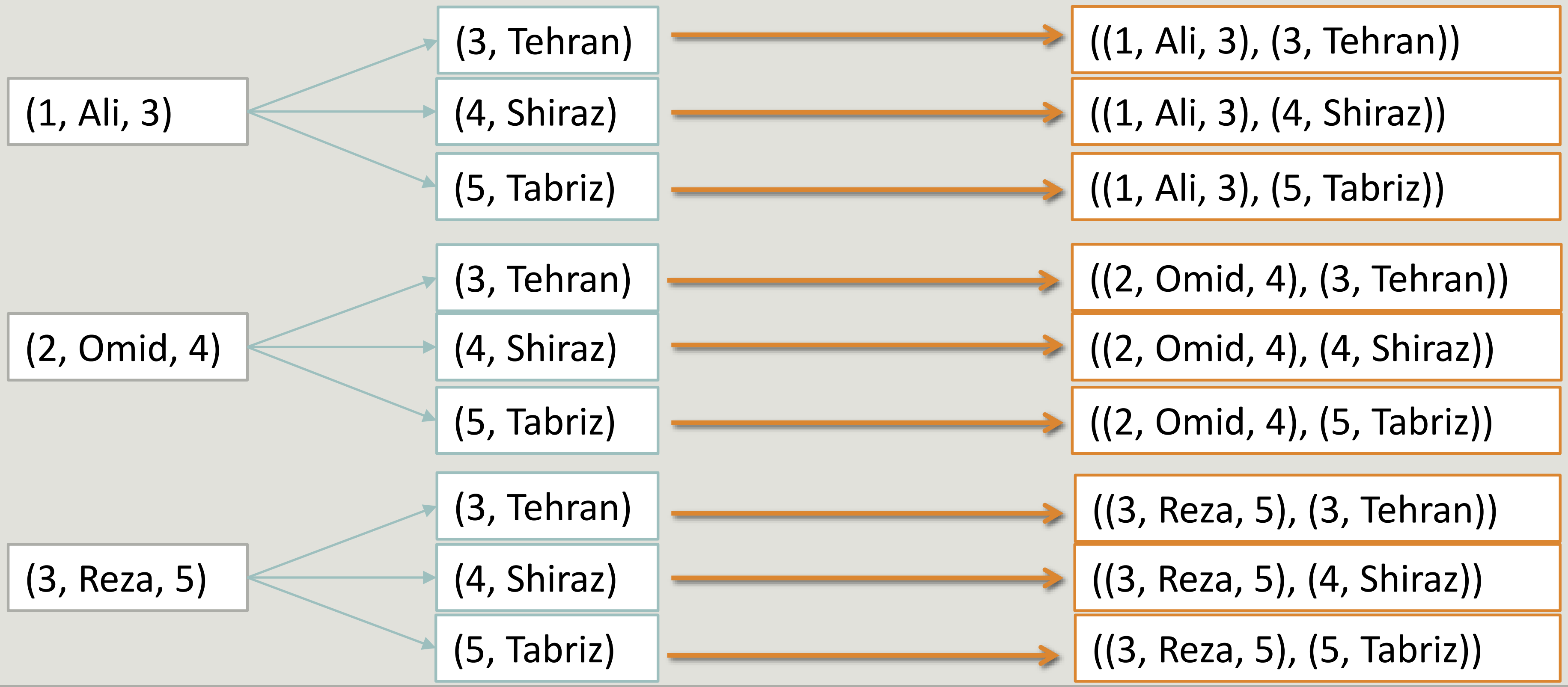
Result			
ID	Name	CityID	CityName
1	Ali	3	Tehran
2	Omid	4	Shiraz
3	Reza	5	Tabriz

```
{       (1, Ali, 3, Tehran),
        (2, Omid, 4, Shiraz),
        (3, Reza, 5, Tabriz)
}
```

(1-J1) Cartesian Product

E = { (1, Ali, 3),   (2, Omid, 4), (3, Reza, 5)**}**
C = { (3, Tehran),   (4, Shiraz),(5, Tabriz)**}**

E * C = ?



(1, Ali, 3)
(3, Tehran)
(4, Shiraz)
(5, Tabriz)
((1, Ali, 3), (3, Tehran))
((1, Ali, 3), (4, Shiraz))
((1, Ali, 3), (5, Tabriz))
(2, Omid, 4)
(3, Tehran)
(4, Shiraz)
(5, Tabriz)
((2, Omid, 4), (3, Tehran))
((2, Omid, 4), (4, Shiraz))
((2, Omid, 4), (5, Tabriz))
(3, Reza, 5)
(3, Tehran)
(4, Shiraz)
(5, Tabriz)
((3, Reza, 5), (3, Tehran))
((3, Reza, 5), (4, Shiraz))
((3, Reza, 5), (5, Tabriz))

(1-J2) On Predicate

Employee			City		ON
ID	Name	CityID	ID	Name	Evaluation
1	Ali	3	3	Tehran	True
1	Ali	3	4	Shiraz	False
1	Ali	3	5	Tabriz	False
2	Omid	4	3	Tehran	False
2	Omid	4	4	Shiraz	True
2	Omid	4	5	Tabriz	False
3	Reza	5	3	Tehran	False
3	Reza	5	4	Shiraz	False
3	Reza	5	5	Tabriz	True

Employee			City	
ID	Name	CityID	ID	Name
1	Ali	3	3	Tehran
2	Omid	4	4	Shiraz
3	Reza	5	5	Tabriz

(1-J3) Add Outer Rows

Employee		
ID	**Name**	**CityID**
1	Ali	3
2	Omid	4

```
{       (1, Ali, 3),
        (2, Omid, 4)
}
```

City	
ID	**Name**
4	Shiraz
5	Tabriz

```
{       (4, Shiraz),
        (5, Tabriz)
}
```

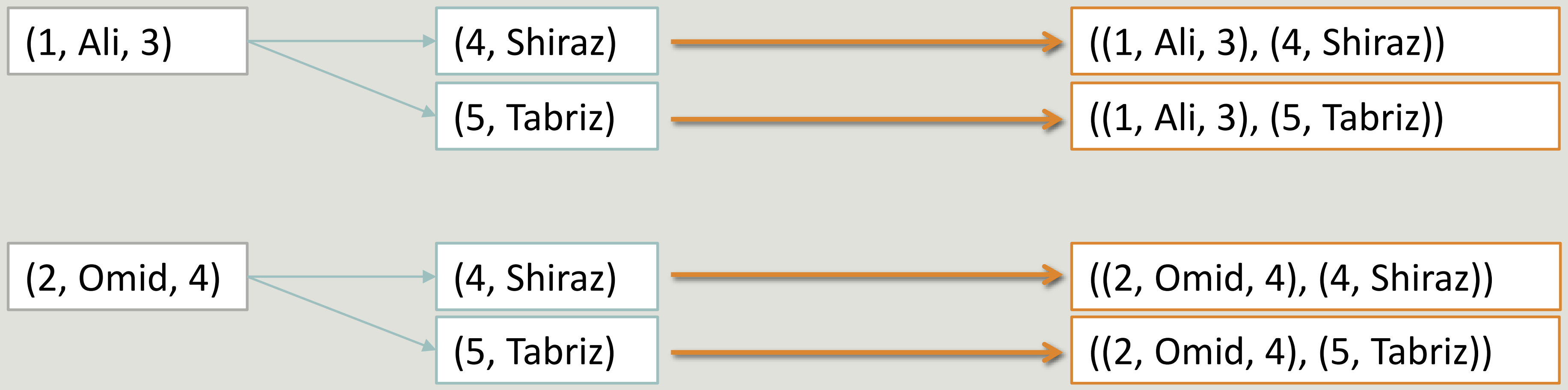
Result			
ID	Name	CityID	CityName
1	Ali	3	NULL
2	Omid	4	Shiraz

```
{       (1, Ali, 3, NULL),
        (2, Omid, 4, Shiraz)
}
```

(1-J3) Add Outer Rows

E = **{** (1, Ali, 3),   (2, Omid, 4)**}**
C = **{** (4, Shiraz),   (5, Tabriz)**}**

E * C = ?



(1, Ali, 3)
(4, Shiraz)
(5, Tabriz)
((1, Ali, 3), (4, Shiraz))
((1, Ali, 3), (5, Tabriz))
(2, Omid, 4)
(4, Shiraz)
(5, Tabriz)
((2, Omid, 4), (4, Shiraz))
((2, Omid, 4), (5, Tabriz))

(1-J3) Add Outer Rows

Employee			City		ON
ID	Name	CityID	ID	Name	**Evaluation**
1	Ali	3	4	Shiraz	**False**
1	Ali	3	5	Tabriz	**False**
2	Omid	4	4	Shiraz	**True**
2	Omid	4	5	Tabriz	**False**

Employee			OUTER ROW
ID	**Name**	**CityID**	**Evaluation**
1	**Ali**	**3**	**True**
2	Omid	4	**False**

City		OUTER ROW
ID	**Name**	**Evaluation**
4	Shiraz	**False**
5	**Tabriz**	**True**

Employee			City	
ID	**Name**	**CityID**	**ID**	**Name**
2	Omid	4	4	Shiraz
1	**Ali**	**3**	**NULL**	**NULL**
NULL	**NULL**	**NULL**	**5**	**Tabriz**

JOINs

JOIN Type		Cartesian Product	On Predicate	Add Outer Rows
CROSS JOIN		OK		
INNER JOIN		OK	OK	
OUTER JOIN	LEFT OUTER JOIN	OK	OK	OK
	RIGHT OUTER JOIN			
	FULL OUTER JOIN			

JOINs

JOIN Type		Old Style	Very Old Style	
CROSS JOIN			,	
INNER JOIN		JOIN	,	=
OUTER JOIN	LEFT OUTER JOIN	LEFT JOIN	,	*=
	RIGHT OUTER JOIN	RIGHT JOIN	,	=*
	FULL OUTER JOIN		,	*=*