JOINING TABLES

SQL

202 – Introduction to Database Systems
Week 4 / Day 1

ITM | systems
IT MUSTAKBIL

LEARNING OBJECTIVES & AGENDA

Learning Objectives:

- Understand how to join relations to de-normalize relations.
- Differentiate between different join types and use appropriate joins in appropriate contexts.

Agenda / Sub-Topics:

- INNER JOIN.
- LEFT JOIN.
- RIGHT JOIN.
- FULL OUTER JOIN.

UNDERSTANDING JOINS

ITM IT MUSTAKBIL | systems

UNDERSTANDING JOINS & DE-NORMALIZATION

- `JOIN` is a command used to connect tuples of one table with tuples from another based on certain conditions.
- The most popular method of finding connecting conditions is to find common / overlapping fields (field names may differ between the two tables).
  - E.g. Selecting a field in one table which is a foreign key that takes its values from a field in the second table.

UNDERSTANDING JOINS & DE-NORMALIZATION

- During normalization, relations are decomposed into smaller relations by building foreign key relationships.
- `JOIN` allows us to reconnect the tuples to re-create the original de-normalized table.
- `JOIN` results in an extended table.

TYPES OF JOINS

ITM IT MUSTAKBIL | systems

INNER JOIN

- Matches tuples from the left table, tuples from the right table wherever the joining condition turns out to be ‘true’.
- If there is a mis-match on either side, both tuples (from both the tables) will be dropped.
- The default join type is the ‘inner join’.

INNER JOIN

Left Table

customers

id	name
1	Ammar Khan
2	Anas Izhar
3	Maryam Tahir
4	Fizza Khan

Joined Table

JOIN

name	order_item
Ammar Khan	LHR-1067
Anas Izhar	KHI-3438
Ammar Khan	LHR-1070
Fizza Khan	ISB-2215

Right Table

customer_orders

customer_id	order_item
1	LHR-1067
2	KHI-3438
1	LHR-1070
4	ISB-2215
5	RWP-1122

- Each tuple from the left is matched with all possible perfect matches from the right.
- Imperfect matches are dropped.

INNER JOIN

```
SELECT *
FROM
   customers AS c
   INNER JOIN customer_orders AS co ON
        c.id = co.customer_id
;
```

LEFT JOIN

- Keeps all tuples from the left table, and tries to match with tuples from the right table wherever possible using the joining condition.
- If a tuple from the left table does not find a match from the right table, the fields taken from the right table are filled with ‘NULL’.

LEFT JOIN

Left Table

customers

id	name
1	Ammar Khan
2	Anas Izhar
3	Maryam Tahir
4	Fizza Khan

Joined Table

JOIN

name	order_item
Ammar Khan	LHR-1067
Ammar Khan	LHR-1070
Anas Izhar	KHI-3438
Maryam Tahir	NULL
Fizza Khan	ISB-2215

Right Table

customer_orders

customer_id	order_item
1	LHR-1067
2	KHI-3438
1	LHR-1070
4	ISB-2215
5	RWP-1122

- All tuples from the left are kept.
- Matching tuples from the right are found where possible.
- All un-matched tuples from the left are assigned ‘NULL’ in fields taken from the right.

LEFT JOIN

```
SELECT *
FROM
   customers AS c
   LEFT JOIN customer_orders AS co ON
        c.id = co.customer_id
;
```

RIGHT JOIN

- Keeps all tuples from the right table, and tries to match with tuples from the left table wherever possible using the joining condition.
- If a tuple from the right table does not find a match from the left table, the fields taken from the left table are filled with ‘NULL’.

RIGHT JOIN

Left Table

customers

id	name
1	Ammar Khan
2	Anas Izhar
3	Maryam Tahir
4	Fizza Khan

Joined Table

JOIN

name	order_item
Ammar Khan	LHR-1067
Anas Izhar	KHI-3438
Ammar Khan	LHR-1070
Fizza Khan	ISB-2215
NULL	RWP-1122

Right Table

customer_orders

customer_id	order_item
1	LHR-1067
2	KHI-3438
1	LHR-1070
4	ISB-2215
5	RWP-1122

- All tuples from the right are kept.
- Matching tuples from the left are found where possible.
- All un-matched tuples from the right are assigned ‘NULL’ in fields taken from the left.

RIGHT JOIN

```
SELECT *
FROM
   customers AS c
   RIGHT JOIN customer_orders AS co ON
        c.id = co.customer_id
;
```