Basic SQL

COMP23111 – Database Systems

Gareth Henshall
Lecturer in Computer Science

The Mario Kart Database

player_name	character_name	rating
Andrea	Toad	5
Gareth	Toad	3
Paul	Bowser	1
Stewart	Mario	2

character_name	size	acceleration	top_speed	handling
Bowser	Large	1	5	2
Luigi	Medium	3	3	2
Mario	Medium	3	3	3
Toad	Small	5	2	4

The SELECT Operation

character_name	size	acceleration	top_speed	handling
Bowser	Large	1	5	2
Luigi	Medium	3	3	2
Mario	Medium	3	3	3
Toad	Small	5	2	4

$\downarrow$

character_name	size	acceleration	top_speed	handling
Bowser	Large	1	5	2
Luigi	Medium	3	3	2

For each character display those where their handling is less than 3

```
SELECT * FROM `Character` WHERE `handling` < 3
```

The PROJECTION Operation

character_name	size	acceleration	top_speed	handling
Bowser	Large	1	5	2
Luigi	Medium	3	3	2
Mario	Medium	3	3	3
Toad	Small	5	2	4

Projection eliminates all attributes of the input relation but those mentioned in the projection list.

character_name	size
Bowser	Large
Luigi	Medium
Mario	Medium
Toad	Small

For each character, display their name and size

```
SELECT `character_name`, `size` FROM `Character`
```

The RENAME Operation

character_name	size	acceleration	top_speed	handling
Bowser	Large	1	5	2
Luigi	Medium	3	3	2
Mario	Medium	3	3	3
Toad	Small	5	2	4

$\downarrow$

character_name	new_name	acceleration	top_speed	handling
Bowser	Large	1	5	2
Luigi	Medium	3	3	2
Mario	Medium	3	3	3
Toad	Small	5	2	4

Rename operation can be used to rename a relation or an attribute of a relation

Change the name of size in the character table to new_name

```
ALTER TABLE `Character` CHANGE `size` `new_name` varchar(20)
```

The UNION Operation

character_name	size	acceleration	top_speed	handling
Bowser	Large	1	5	2
Luigi	Medium	3	3	2
Mario	Medium	3	3	3
Toad	Small	5	2	4

player_name	character_name	rating
Andrea	Toad	5
Gareth	Toad	3
Paul	Bowser	1
Stewart	Mario	2

character_name
Bowser
Luigi
Mario
Toad

It includes all tuples that are in tables A or in B. It also eliminates duplicate tuples.

The following must be valid:

1. R_1 and R_2 must have the same number of attributes
2. Attribute domains need to be compatible
3. Duplicate tuples should be automatically removed

Find duplicates of character_names

```
SELECT `character_name` FROM `Character`
UNION SELECT `character_name` FROM `Player`
```

The CARTESIAN PRODUCT Operation

character_name	size	acceleration	top_speed	handling
Bowser	Large	1	5	2
Luigi	Medium	3	3	2
Mario	Medium	3	3	3
Toad	Small	5	2	4

character_name	size	acceleration	top_speed	handling	player_name	character_name	rating
Bowser	Large	1	5	2	Andrea	Toad	5
Luigi	Medium	3	3	2	Andrea	Toad	5
Mario	Medium	3	3	3	Andrea	Toad	5
Toad	Small	5	2	4	Andrea	Toad	5
Bowser	Large	1	5	2	Gareth	Toad	3
Luigi	Medium	3	3	2	Gareth	Toad	3
Mario	Medium	3	3	3	Gareth	Toad	3
Toad	Small	5	2	4	Gareth	Toad	3
Bowser	Large	1	5	2	Paul	Bowser	1
Luigi	Medium	3	3	2	Paul	Bowser	1
Mario	Medium	3	3	3	Paul	Bowser	1
Toad	Small	5	2	4	Paul	Bowser	1
Bowser	Large	1	5	2	Stewart	Mario	2
Luigi	Medium	3	3	2	Stewart	Mario	2
Mario	Medium	3	3	3	Stewart	Mario	2
Toad	Small	5	2	4	Stewart	Mario	2

This type of operation is helpful to merge columns from two relations.

Generally, a Cartesian product is never a meaningful operation when it performs alone.

However, it becomes meaningful when it is followed by other operations

Combine and display everything from both tables

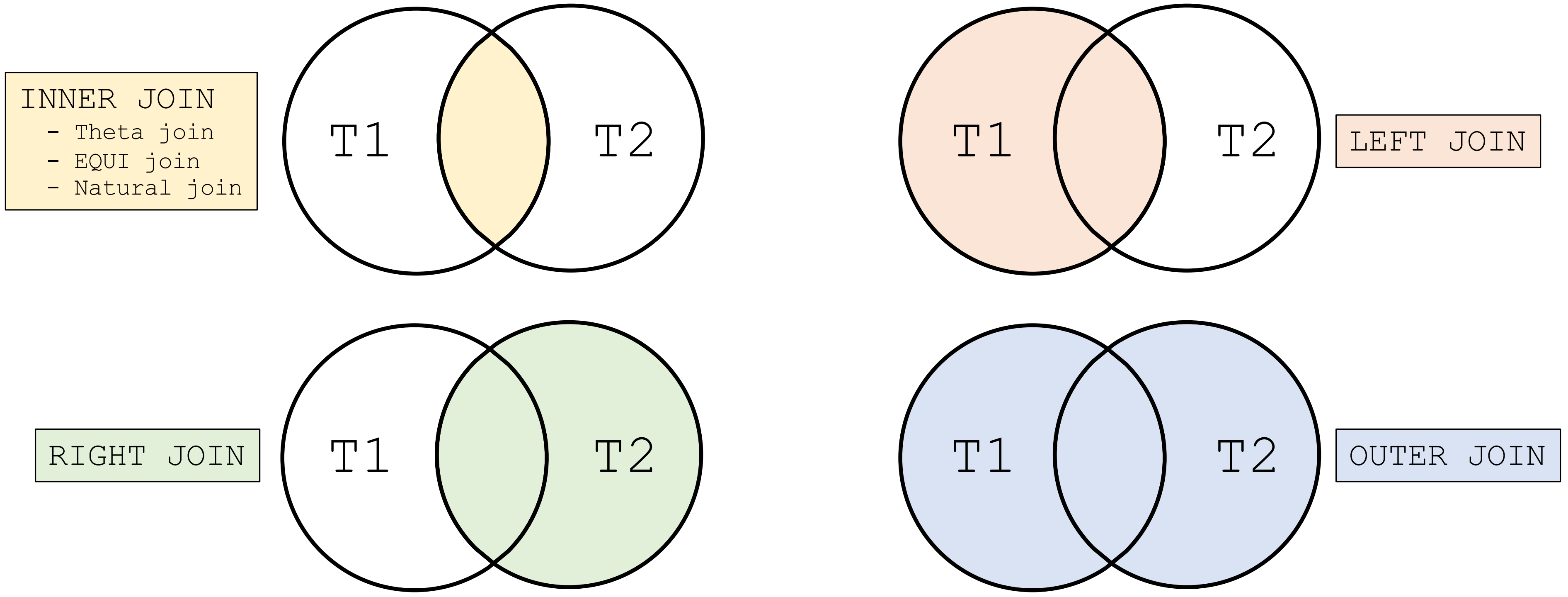
```
SELECT * FROM `Character` CROSS JOIN `Player`
```

The CARTESIAN PRODUCT Operation

character_name	size	acceleration	top_speed	handling	player_name	character_name	rating
Bowser	Large	1	5	2	Andrea	Toad	5
Luigi	Medium	3	3	2	Andrea	Toad	5
Mario	Medium	3	3	3	Andrea	Toad	5
Toad	Small	5	2	4	Andrea	Toad	5
Bowser	Large	1	5	2	Gareth	Toad	3
Luigi	Medium	3	3	2	Gareth	Toad	3
Mario	Medium	3	3	3	Gareth	Toad	3
Toad	Small	5	2	4	Gareth	Toad	3
Bowser	Large	1	5	2	Paul	Bowser	1
Luigi	Medium	3	3	2	Paul	Bowser	1
Mario	Medium	3	3	3	Paul	Bowser	1
Toad	Small	5	2	4	Paul	Bowser	1
Bowser	Large	1	5	2	Stewart	Mario	2
Luigi	Medium	3	3	2	Stewart	Mario	2
Mario	Medium	3	3	3	Stewart	Mario	2
Toad	Small	5	2	4	Stewart	Mario	2

```
SELECT * FROM `Character` CROSS JOIN `Player`
```

Types of JOIN



The THETA Join

character_name	size	acceleration	top_speed	handling
Bowser	Large	1	5	2
Luigi	Medium	3	3	2
Mario	Medium	3	3	3
Toad	Small	5	2	4

Theta join can use any conditions in the selection criteria

Display entries where a character's acceleration is greater than a player's rating

player_name	character_name	rating
Andrea	Toad	5
Gareth	Toad	3
Paul	Bowser	1
Stewart	Mario	2

character_name	size	acceleration	top_speed	handling	player_name	rating
Toad	Small	5	2	4	Gareth	3
Luigi	Medium	3	3	2	Paul	1
Mario	Medium	3	3	3	Paul	1
Toad	Small	5	2	4	Paul	1
Luigi	Medium	3	3	2	Stewart	2
Mario	Medium	3	3	3	Stewart	2
Toad	Small	5	2	4	Stewart	2

```
SELECT * FROM `Character`, `Player`
WHERE `Character`.`acceleration` > `Player`.`rating`
```

The EQUI Join

character_name	size	acceleration	top_speed	handling
Bowser	Large	1	5	2
Luigi	Medium	3	3	2
Mario	Medium	3	3	3
Toad	Small	5	2	4

When a theta join uses only equivalence condition, it becomes a equi join.

Display entries where a character's acceleration is equal to a player's rating

player_name	character_name	rating
Andrea	Toad	5
Gareth	Toad	3
Paul	Bowser	1
Stewart	Mario	2

character_name	size	acceleration	top_speed	handling	player_name	rating
Toad	Small	5	2	4	Andrea	5
Luigi	Medium	3	3	2	Gareth	3
Mario	Medium	3	3	3	Gareth	3
Bowser	Large	1	5	2	Paul	1

```
SELECT * FROM `Character`, `Player`
WHERE `Character`.`acceleration` = `Player`.`rating`
```

The NATURAL Join

character_name	size	acceleration	top_speed	handling
Bowser	Large	1	5	2
Luigi	Medium	3	3	2
Mario	Medium	3	3	3
Toad	Small	5	2	4

Natural join can only be performed if there is a common attribute (column) between the relations.

The name and type of the attribute must be same.

Join the character and player tables and display the results

player_name	character_name	rating
Andrea	Toad	5
Gareth	Toad	3
Paul	Bowser	1
Stewart	Mario	2

character_name	size	acceleration	top_speed	handling	player_name	character_name	rating
Toad	Small	5	2	4	Andrea	Toad	5
Toad	Small	5	2	4	Gareth	Toad	3
Bowser	Large	1	5	2	Paul	Bowser	1
Mario	Medium	3	3	3	Stewart	Mario	2

```
SELECT * FROM `Character` NATURAL JOIN `Player`
```

The LEFT Join

character_name	size	acceleration	top_speed	handling
Bowser	Large	1	5	2
Luigi	Medium	3	3	2
Mario	Medium	3	3	3
Toad	Small	5	2	4

In the left join, operation allows keeping all tuple in the left relation.

However, if there is no matching tuple is found in right relation, then the attributes of right relation in the join result are filled with null values.

player_name	character_name	rating
Andrea	Toad	5
Gareth	Toad	3
Paul	Bowser	1
Stewart	Mario	2

character_name	size	acceleration	top_speed	handling	player_name	character_name	rating
Toad	Small	5	2	4	Andrea	Toad	5
Toad	Small	5	2	4	Gareth	Toad	3
Bowser	Large	1	5	2	Paul	Bowser	1
Mario	Medium	3	3	3	Stewart	Mario	2
Luigi	Medium	3	3	2	(null)	(null)	(null)

```
SELECT * FROM `Character` LEFT JOIN `Player`
ON `Character`.`character_name` = `Player`.`character_name`
```

The RIGHT Join

character_name	size	acceleration	top_speed	handling
Bowser	Large	1	5	2
Luigi	Medium	3	3	2
Mario	Medium	3	3	3
Toad	Small	5	2	4

player_name	character_name	rating
Andrea	Toad	5
Gareth	Toad	3
Paul	Bowser	1
Stewart	Mario	2

character_name	size	acceleration	top_speed	handling	player_name	rating
Toad	Small	5	2	4	Andrea	5
Toad	Small	5	2	4	Gareth	3
Bowser	Large	1	5	2	Paul	1
Mario	Medium	3	3	3	Stewart	2

In the right join, operation allows keeping all tuple in the right relation.

However, if there is no matching tuple is found in the left relation, then the attributes of the left relation in the join result are filled with null values.

```
SELECT * FROM `Character` RIGHT JOIN `Player`
ON `Character`.`character_name` = `Player`.`character_name`
```

More Complex SQL Queries

character_name	size	acceleration	top_speed	handling
Bowser	Large	1	5	2
Luigi	Medium	3	3	2
Mario	Medium	3	3	3
Toad	Small	5	2	4

player_name	character_name	rating
Andrea	Toad	5
Gareth	Toad	3
Paul	Bowser	1
Stewart	Mario	2

character_name	player_name
Bowser	Andrea
Luigi	Andrea
Mario	Andrea
Bowser	Gareth
Bowser	Stewart

Display all unique instances the characters name and players name where the characters acceleration is less than the players rating

```
SELECT DISTINCT `Character`.`character_name`, `Player`.`player_name`
FROM `Character`, `Player` WHERE `Character`.`acceleration` < `Player`.`rating`
```

More Complex SQL Queries

character_name	size	acceleration	top_speed	handling
Bowser	Large	1	5	2
Luigi	Medium	3	3	2
Mario	Medium	3	3	3
Toad	Small	5	2	4

player_name	character_name	rating
Andrea	Toad	5
Gareth	Toad	3
Paul	Bowser	1
Stewart	Mario	2

character_name	handling	player_name	rating
Bowser	2	Andrea	5
Bowser	2	Gareth	3
Luigi	2	Andrea	5
Luigi	2	Gareth	3
Mario	3	Andrea	5
Toad	4	Andrea	5

Join the two tables and display all unique instances of the characters name & handling and players name & rating where the characters handling is less than the players rating

```
SELECT DISTINCT `Character`.`character_name`, `Character`.`handling`,
`Player`.`player_name`, `Player`.`rating`
FROM `Player` INNER JOIN `Character` ON `Character`.`handling` < `Player`.`rating`;
```

More Complex SQL Queries

player_name	character_name	rating
Andrea	Toad	5
Gareth	Toad	3
Paul	Bowser	1
Stewart	Mario	2

→

WorstPlayer
3

Display the rating of the player who selected toad with the worst rating in a column headed WorstPlayer

```
SELECT DISTINCT MIN(`Player`.`rating`) AS `WorstPlayer`
FROM `Player` WHERE `Player`.`character_name` = 'Toad';
```