Engineering Databases

Lecture 5 – Aggregation, Grouping, and Nesting

November 16, 2022

M. Saeed Mafipour & Mansour Mehranfar

Contents of Lecture 4

- Foreign key integrity constraint
  - A essential principle in relational design
  - Ensures that a value in a table is contained in some other table
  - Control updates and changes of other tables
- Relational algebra and joins
  - Cartesian product $\times$
  - Renaming ρ
  - θ-Join $\bowtie_\theta$ and equi-Join $\bowtie_{\theta\ is\ =}$
  - (Natural) Join $\bowtie$
  - Right $\ltimes$ and left semi join $\rtimes$
  - Left $⟕$, right $⟖$, and full outer join $⟗$

Explicit Renaming

- Remember: Renaming a relation/table <relation> as <new Name>
- Renaming an attribute/column
  - Operator: $\rho_{new\ name/old\ name}(relation)$
  - The SQL syntax for the renaming an attribute/column:
    SELECT <attribute> AS <new Name> FROM <Relation>
  - Example:
    SELECT number AS roomNumber, building FROM rooms

rooms	
number	building
2	N1
3	N4

$\rho_{roomNumber/number}(rooms)$

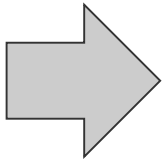
$\rho_{number/roomNumber}(rooms)$	
roomNumber	building
2	N1
3	N4

  - Exercise: $\rho_{structure/building}(rooms)$

Aggregate Functions and Grouping

- Aggregated group entries
- The SQL syntax for the grouping based on an attribute/column:
  SELECT <attributes> FROM <Relation>
  GROUP BY <attribute>
- Example: SELECT firstName FROM persons GROUP BY firstName

firstName	lastName	age	male
Max	Bügler	33	1
Max	Mustermann	20	1
Max	Müller	25	1
Parker	James	28	0
Parker	Miller	24	1
Jennifer	Milan	22	0
Jennifer	Turner	28	0
John	Turner	25	1
Jennifer	Turner	21	0



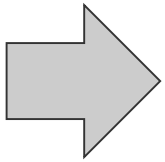
firstName
Jennifer
John
Max
Parker

https://dev.mysql.com/doc/refman/5.7/en/group-by-functions.html

Aggregate Functions and Grouping

- Aggregate grouped entries by columns containing the same values
- The SQL syntax for aggregating group attributes:
  SELECT <groupAttribute>, <function on aggregated Attributes>
  FROM <Relation> GROUP BY <groupAttribute>
- Example: SELECT firstName, count(lastName) AS counts
  FROM persons GROUP BY firstName

firstName	lastName	age	male
Max	Bügler	33	1
Max	Mustermann	20	1
Max	Müller	25	1
Parker	James	28	0
Parker	Miller	24	1
Jennifer	Milan	22	0
Jennifer	Turner	28	0
John	Turner	25	1
Jennifer	Turner	21	0



firstName	counts
Jennifer	3
John	1
Max	3
Parker	2

https://dev.mysql.com/doc/refman/5.7/en/group-by-functions.html

Aggregate Functions and Grouping

- Different aggregation functions
- Can be combined by OR, AND, NOT → CONCAT(“EngDB”, “CMS”) → “EngDBCMS”

AVG()	Average Value
COUNT()	Number of (DISTINCT) rows
GROUP_CONCAT()	Concatenate Strings
MAX(), MIN()	Max/min Values
STDDEV()	Standard Deviation of values
SUM()	Sum of values
BIT_AND(), BIT_OR()	Bitwise AND/OR
…	

https://dev.mysql.com/doc/refman/5.7/en/group-by-functions.html

firstName	lastName	age	male
Max	Bügler	33	1
Max	Mustermann	20	1
Max	Müller	25	1
Parker	James	28	0
Parker	Miller	24	1
Jennifer	Milan	22	0
Jennifer	Turner	28	0
John	Turner	25	1
Jennifer	Turner	21	0

Aggregate Functions and Grouping

```
SELECT
  firstName,
  GROUP_CONCAT(lastName) AS lastNames,
  GROUP_CONCAT(DISTINCT lastName) AS distinctLastNames,
  COUNT(lastName) AS count,
  COUNT(DISTINCT lastname) AS distinctCount,
  AVG(age) AS averageAge,
  STDDEV(age) AS ageStDev,
  BIT_OR(male) AS containsMales,
  NOT BIT_AND(male) AS containsFemales,
  BIT_OR(male) AND NOT BIT_AND(male) AS unisexName
FROM Persons
GROUP BY firstName;
```

firstName
Jennifer
John
Max
Parker

https://dev.mysql.com/doc/refman/5.7/en/group-by-functions.html

Aggregate Functions and Grouping - HAVING

firstName	lastName	age	male
Max	Bügler	33	1
Max	Mustermann	20	1
Max	Müller	25	1
Parker	James	28	0
Parker	Miller	24	1
Jennifer	Milan	22	0
Jennifer	Turner	28	0
John	Turner	25	1
Jennifer	Turner	21	0

- Apply conditions for a group of rows or aggregates
- The SQL syntax for aggregating group attributes:
  SELECT <groupAttribute>
  FROM <Relation>
  GROUP BY <groupAttribute>
  HAVING <condition>

- Example:
  SELECT firstName FROM Persons GROUP BY firstName HAVING COUNT(lastName)>1;

firstName
Jennifer
Max
Parker

  SELECT firstName, GROUP_CONCAT(Age) AS ages
  FROM Persons GROUP BY firstName
  HAVING SUM(Age)>60

firstName	ages
Jennifer	22,28,21
Max	33,20,25

https://dev.mysql.com/doc/refman/5.7/en/group-by-handling.html

Nested Queries and Quantors

- Combine nested queries with WHERE statements
- Example:

SELECT * FROM Persons **WHERE** age>30

firstName	lastName	age	male
Max	Bügler	33	1
Max	Mustermann	20	1
Max	Müller	25	1
Parker	James	28	0
Parker	Miller	24	1
Jennifer	Milan	22	0
Jennifer	Turner	28	0
John	Turner	25	1
Jennifer	Turner	21	0

SELECT * FROM Persons **WHERE** age > (SELECT AVG(age) FROM Persons);

Returns a single value

firstName	lastName	age	male
Max	Bügler	33	1
Parker	James	28	0
Jennifer	Turner	28	0

https://dev.mysql.com/doc/refman/5.7/en/subqueries.html

Nested Queries and Quantors - EXISTS

VIPs

lastName
Turner
Milan

firstName	lastName	age	male
Max	Bügler	33	1
Max	Mustermann	20	1
Max	Müller	25	1
Parker	James	28	0
Parker	Miller	24	1
Jennifer	Milan	22	0
Jennifer	Turner	28	0
John	Turner	25	1
Jennifer	Turner	21	0

- List entries for which another query returns something
- **EXISTS** quantor
- Example:

SELECT * FROM Persons WHERE **EXISTS**

(SELECT * FROM VIPs WHERE VIPs.lastName=Persons.lastName);

firstName	lastName	age	male
Jennifer	Milan	22	0
Jennifer	Turner	28	0
John	Turner	25	1
Jennifer	Turner	21	0

SELECT * FROM Persons WHERE **NOT EXISTS**

(SELECT * FROM VIPs WHERE VIPs.lastName=Persons.lastName);

firstName	lastName	age	male
Max	Bügler	33	1
Max	Mustermann	20	1
Max	Müller	25	1
Parker	James	28	0
Parker	Miller	24	1

https://dev.mysql.com/doc/refman/5.7/en/exists-and-not-exists-subqueries.html

Nested Queries and Quantors - EXISTS

VIPs

lastName
Turner
Milan

firstName	lastName	age	male
Max	Bügler	33	1
Max	Mustermann	20	1
Max	Müller	25	1
Parker	James	28	0
Parker	Miller	24	1
Jennifer	Milan	22	0
Jennifer	Turner	28	0
John	Turner	25	1
Jennifer	Turner	21	0

- Define whether another query returns something as a boolean column using the EXISTS quantor.
- Example:

```
SELECT firstName, lastName,
(EXISTS
        (SELECT * FROM VIPs WHERE VIPs.lastName=Persons.lastName))
AS isVIP
FROM Persons;
```

firstName	lastName	isVIP
Max	Bügler	0
Max	Mustermann	0
Max	Müller	0
Parker	James	0
Parker	Miller	0
Jennifer	Milan	1
Jennifer	Turner	1
John	Turner	1
Jennifer	Turner	1

https://dev.mysql.com/doc/refman/5.7/en/exists-and-not-exists-subqueries.html

firstName	lastName	age	male
Max	Bügler	33	1
Max	Mustermann	20	1
Max	Müller	25	1
Parker	James	28	0
Parker	Miller	24	1
Jennifer	Milan	22	0
Jennifer	Turner	28	0
John	Turner	25	1
Jennifer	Turner	21	0

Special Language Elements

- Get values in a certain range using **BETWEEN**
- Example:
  SELECT * FROM Persons WHERE age BETWEEN 20 AND 25

firstName	lastName	age	male
Max	Mustermann	20	1
Max	Müller	25	1
Parker	Miller	24	1
Jennifer	Milan	22	0
John	Turner	25	1
Jennifer	Turner	21	0

SELECT * FROM Persons WHERE age IN (11,22,33);

firstName	lastName	age	male
Max	Bügler	33	1
Jennifer	Milan	22	0

firstName	lastName	age	male
Max	Bügler	33	1
Max	Mustermann	20	1
Max	Müller	25	1
Parker	James	28	0
Parker	Miller	24	1
Jennifer	Milan	22	0
Jennifer	Turner	28	0
John	Turner	25	1
Jennifer	Turner	21	0

Special Language Elements

- Match string patterns using **LIKE**
- Any number of characters matches a **%**
- Example:
  SELECT * FROM Persons WHERE lastName LIKE "m%";

firstName	lastName	age	male
Max	Mustermann	20	1
Max	Müller	25	1
Parker	Miller	24	1
Jennifer	Milan	22	0

SELECT * FROM Persons WHERE lastName LIKE "%m%";

firstName	lastName	age	male
Max	Mustermann	20	1
Max	Müller	25	1
Parker	James	28	0
Parker	Miller	24	1
Jennifer	Milan	22	0

https://dev.mysql.com/doc/refman/5.7/en/pattern-matching.html

firstName	lastName	age	male
Max	Bügler	33	1
Max	Mustermann	20	1
Max	Müller	25	1
Parker	James	28	0
Parker	Miller	24	1
Jennifer	Milan	22	0
Jennifer	Turner	28	0
John	Turner	25	1
Jennifer	Turner	21	0

Special Language Elements

- Match string patterns using **LIKE**
- A single character matches _
- Example:
  SELECT * FROM Persons WHERE lastName LIKE "M_ller";

firstName	lastName	age	male
Max	Müller	25	1
Parker	Miller	24	1

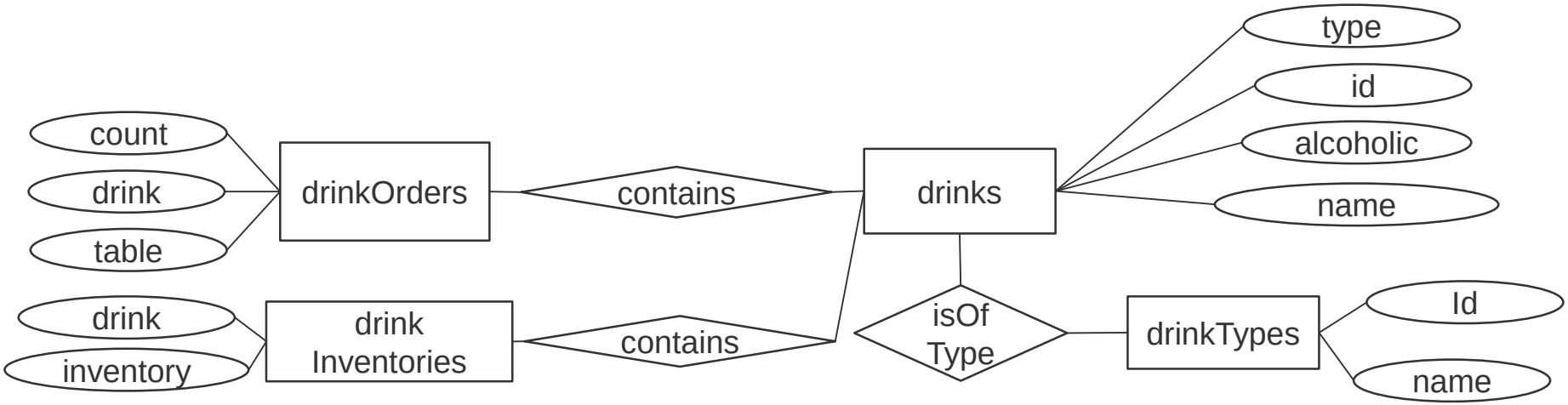
SELECT * FROM Persons WHERE lastName LIKE "_____";

firstName	lastName	age	male
Parker	James	28	0
Jennifer	Milan	22	0

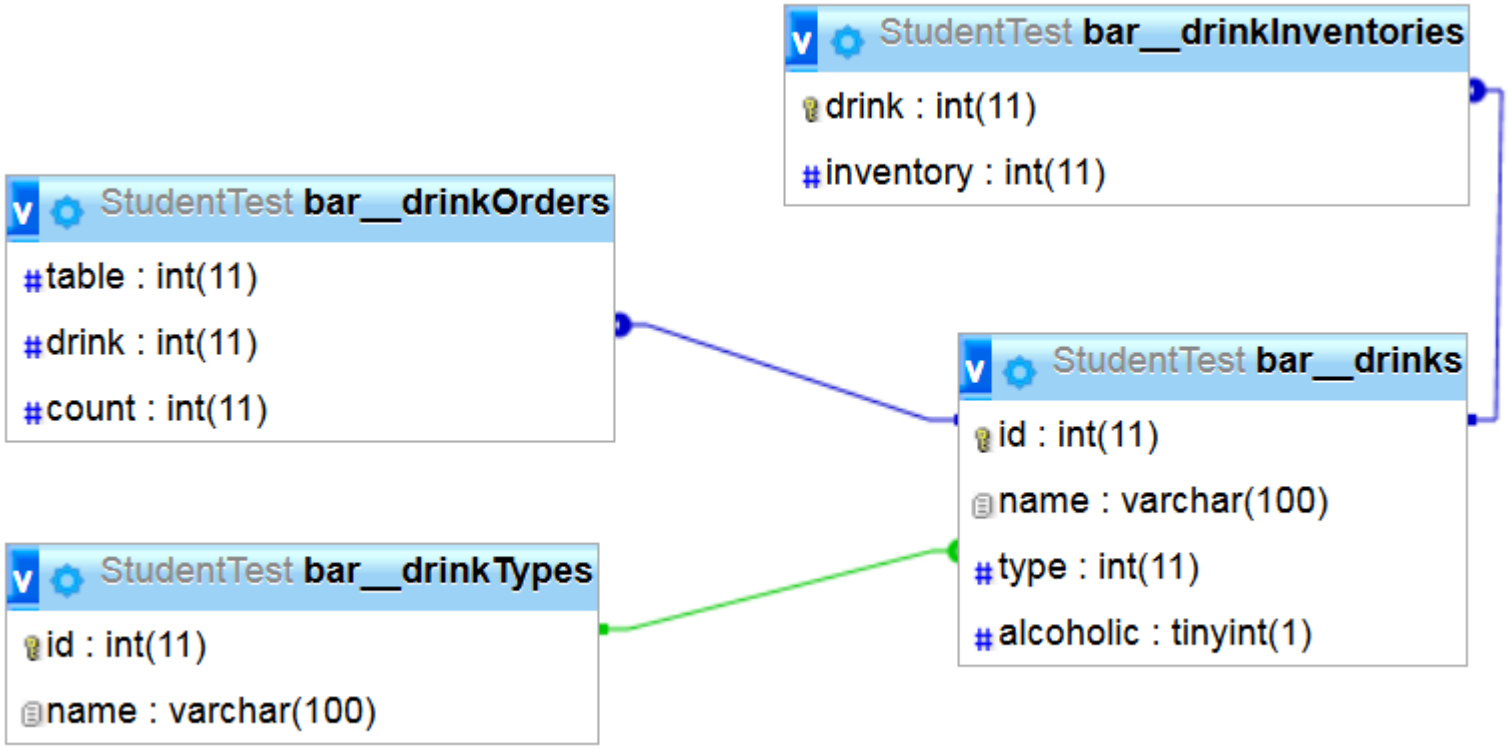
https://dev.mysql.com/doc/refman/5.7/en/pattern-matching.html

Exercise bar example

- Use the bar_generate file from moodle and create the tables and insert the data.
- Carefully review (and finalize) the ER-Diagram. (N:1/M?, primary key?)
- Carefully review the SQL DDL statements.
- Write down all SELECT statements.



Bar tables and foreign keys



Homework, bar example, select

- List all drink names
- List all non alcoholic drinks
- List all drinks and how many of them are in inventory
- List all drinks that are in the inventory and how many of them are in the inventory
- List all non alcoholic drinks that are in the inventory and how many of them are in the inventory
- List all cocktails that are in the inventory and List all cocktails that are in stock
- List all cocktails that are the inventory, but rename column to „Cocktail“
- List Tables that have ordered a cocktail that is in the inventory and what cocktail was ordered
- List all drinks ordered and the table they were ordered to. Mark drinks that were ordered by no one by NULL.
- List all drinks that were ordered
- List all drinks that were not ordered
- List all beers that were not ordered
- How many drinks were ordered for each type?
- List all cocktails and beers

End of Lecture

Thank you for your attention