Querying

Holger Pirk

Some housekeeping first

We've release the first coursework today

- it is due in exactly two weeks (November 3)
- the focus is on ORM, SQL and storage

You will need to use a VM on the cloudstack cluster

- Create it using the web interface
- Log in using ssh
- You can develop on the VM or locally
- Solutions will be marked against the VM

A brief demo

- How to set up a VM

More housekeeping

Supervised labtime

- Next Wednesday and Friday at 1pm in 202/206
- C++ is a b***h, let us help you

Haven't provided answers for Monday's calculations

- Sorry
- Will do it soon

Purpose of this lecture

What you know:

- How to design redundancy-free schemas
- How to get data into the DBMS using ORMs
- How data is stored internaly

Today you will learn how to get it out

- Relational Algebra
- Basic SQL
  - Enough to get you through the coursework
  - We will not talk about joins

The Running Example

Customer

ID	Name	ShippingAddress
1	Holger	180 Queens Gate
2	Sam	32 Vassar Street
3	Peter	180 Queens Gate

Order

ID	CustomerID
1	1
2	2
3	3

OrderedItem

OrderId	BookID
1	1
1	2
2	1
3	3

Book

ID	Title	Author
1	Database Management Systems	Ramakrishnan & Gehrke
2	A Game of Thrones	Martin
3	Distributed Systems	van Steen & Tanenbaum

Relational Algebra

The Purpose of Relational Algebra

A logical representation of relational operations

- Used to define the semantics of operations
- Used for logical optimization
- Not actually all that useful for end-users

Fundamentals

Nomenclatures

- An expression in relational algebra is composed from operators
- I will often refer to an expression as a plan
- Sometimes as a logical plan
- Cardinality is the number of tuples in a set

Handy properties

- Set-based
  - Order-invariant and duplicate eliminated
- Relational algebra is closed
  - Every operator produces a relation as output
  - Every operator accepts one or two relations as input
  - This simplifies the composition of operators into expressions
    - Note, that expressions can still be invalid

Relational Operators

A minimal set

- Select
- Project
- Cross (Carthesian) Product
- Union
- Difference

Not included

- Intersection

Project π

Intuitive semantics

- Extract one or multiple attributes from a relation
- Preserve relational semantics
- Changes the schema

Customer

ID	Name	ShippingAddress
1	Holger	180 Queens Gate
2	Sam	32 Vassar Street
3	Peter	180 Queens Gate

π_{Name} Customer

Name
Holger
Sam
Peter

$\pi_{ShippingAddress}$ Customer

ShippingAddress
180 Queens Gate
32 Vassar Street

Project π

Quick Quiz: What is the cardinality of the output of a projection

- It can only be determined by evaluating it
- Cardinality of the input
- I don't know

Project π

Quick Quiz: What is the cardinality of the output of a projection

- It can only be determined by evaluating it
- Cardinality of the input
- I don't know

Quick Quiz: What is the upper bound for the cardinality of the output of a project

- It can only be determined by evaluating it
- Cardinality of the input
- I don't know

Select σ

Intuitive Semantics

- Produce a new relation containing input tuples that satisfy a condition
- Does not change the schema
- Changes cardinality (i.e., the number of tuples in a relation)

OrderedItem

OrderId	BookID
1	1
1	2
2	1
3	3

$\sigma_{BookID=1} OrderedItem$

OrderId	BookID
1	1
2	1

Select σ

Quick Quiz: What is the cardinality of the output of a selection

- It can only be determined by evaluating it
- Cardinality of the input
- I don't know

Select σ

Quick Quiz: What is the cardinality of the output of a selection

- It can only be determined by evaluating it
- Cardinality of the input
- I don't know

Quick Quiz: What is the upper bound for the cardinality of the output of a select

- It can only be determined by evaluating it
- Cardinality of the input
- I don't know

Cross (Carthesian) Product $\times$

Order

ID	CustomerID
1	1
2	2
3	3

OrderedItem

OrderId	BookID
1	1
1	2
2	1
3	3

Order $\times$ *OrderedItem*

ID	CustomerID	OrderId	BookID
1	1	1	1
1	1	1	2
1	1	2	1
1	1	3	3
2	2	1	1
2	2	1	2
2	2	2	1
2	2	3	3
3	3	1	1
3	3	1	2
3	3	2	1
3	3	3	3

Cross (Carthesian) Product $\times$

Intuitive Semantics

- Takes two inputs
- Produce a new relation by combining every tuple from the left with every tuple from the right
- Changes the schema

Cross (Carthesian) Product $\times$

Intuitive Semantics

- Takes two inputs
- Produce a new relation by combining every tuple from the left with every tuple from the right
- Changes the schema

Quick Quiz: What is the cardinality of the output of a cross product

- It can only be determined by evaluating it
- Cardinality of left input plus cardinality of right input
- Cardinality of left input times cardinality of right input
- I don't know

Composing operators

Rules

- Since relational algebra is closed, operators can be combined as long as their signature is respected
  - Cross products take two inputs
  - Selections and Projections take one

Example

$\pi_{BookID}(\sigma_{ID==OrderID}(Order \times OrderedItem))$

Worksheet

Customer

ID	Name	ShippingAddress
1	Holger	180 Queens Gate
2	Sam	32 Vassar Street
3	Peter	180 Queens Gate

Order

ID	CustomerID
1	1
2	2
3	3

OrderedItem

OrderId	BookID
1	1
1	2
2	1
3	3

Book

ID	Title	Author
1	Database Management Systems	Ramakrishnan & Gehrke
2	A Game of Thrones	Martin
3	Distributed Systems	van Steen & Tanenbaum

The task

- From the operators we have discussed, compose a query that finds the titles of all books I (holger) have ordered.

Worksheet

Customer

ID	Name	ShippingAddress
1	Holger	180 Queens Gate
2	Sam	32 Vassar Street
3	Peter	180 Queens Gate

Order

ID	CustomerID
1	1
2	2
3	3

OrderedItem

OrderId	BookID
1	1
1	2
2	1
3	3

Book

ID	Title	Author
1	Database Management Systems	Ramakrishnan & Gehrke
2	A Game of Thrones	Martin
3	Distributed Systems	van Steen & Tanenbaum

The task

- From the operators we have discussed, compose a query that finds the titles of all books I (holger) have ordered.

Congratulations

- You have performed a join

Union $\cup$

Intuitive Semantics

- Produce a new relation from two relations containing any tuple that is present in one of the inputs
- Does not change the schema
  - Requires union compatibility
- Changes cardinality (i.e., the number of tuples in a relation)

$\sigma_{Name=Holger}\, Customer$

ID	Name	ShippingAddress
1	Holger	180 Queens Gate

$\sigma_{Name=Sam}\, Customer$

ID	Name	ShippingAddress
2	Sam	32 Vassar Street

$\sigma_{Name=Sam}\, Customer \cup \sigma_{Name=Holger}\, Customer$

ID	Name	ShippingAddress
1	Holger	180 Queens Gate
2	Sam	32 Vassar Street

Quick quiz: Union $\cup$

Quick Quiz: What is the cardinality of the output of a union

- It can only be determined by evaluating it
- Cardinality of left input plus cardinality of right input
- Cardinality of left input times cardinality of right input
- I don't know

Quick quiz: Union $\cup$

Quick Quiz: What is the cardinality of the output of a union

- It can only be determined by evaluating it
- Cardinality of left input plus cardinality of right input
- Cardinality of left input times cardinality of right input
- I don't know

Quick Quiz: What is the upper bound for the cardinality of a union

- It can only be determined by evaluating it
- Cardinality of left input plus cardinality of right input
- Cardinality of left input times cardinality of right input
- I don't know

Difference −

Intuitive Semantics

- Produce a new relation from two relations containing any tuple that is present in the first but not the second input
- Does not change the schema
  - Requires union compatibility
- Changes cardinality (i.e., the number of tuples in a relation)

Customer

ID	Name	ShippingAddress
1	Holger	180 Queens Gate
2	Sam	32 Vassar Street
3	Peter	180 Queens Gate

$\sigma_{Name=Sam}$ *Customer*

ID	Name	ShippingAddress
2	Sam	32 Vassar Street

Customer − $\sigma_{Name=Sam}$ *Customer*

ID	Name	ShippingAddress
1	Holger	180 Queens Gate
3	Peter	180 Queens Gate

The Structured Query Language

SQL

Purpose

- Not a faithful implementation of relational algebra
- Meant to be useful
- Unfortunately, the S is not for "standard"

A language people like to hate

- An anecdote about SQL

Our example query

Customer

ID	Name	ShippingAddress
1	Holger	180 Queens Gate
2	Sam	32 Vassar Street
3	Peter	180 Queens Gate

Order

ID	CustomerID
1	1
2	2
3	3

OrderedItem

OrderId	BookID
1	1
1	2
2	1
3	3

Book

ID	Title	Author
1	Database Management Systems	Ramakrishnan & Gehrke
2	A Game of Thrones	Martin
3	Distributed Systems	van Steen & Tanenbaum

A mildly complex query

```
select book.title as t from customer, book, order,
orderedItem
where customer.name = "holger"
and book.id = orderedItem.BookID
and order.id = orderedItem.orderID
and customer.id = order.customerId
```

Selections in SQL

Mapping

```
select book.title as t from customer, book, order,
orderedItem
where customer.name = "holger"
and book.id = orderedItem.BookID
and order.id = orderedItem.orderID
and customer.id = order.customerId
```

Projections in SQL

Mapping

```
select book.title as t from customer, book, order,
orderedItem
where customer.name = "holger"
and book.id = orderedItem.BookID
and order.id = orderedItem.orderID
and customer.id = order.customerId
```

Cross (Carthesian) Product in SQL

Mapping

```
select book.title as t from customer, book, order,
orderedItem
where customer.name = "holger"
and book.id = orderedItem.BookID
and order.id = orderedItem.orderID
and customer.id = order.customerId
```

Set Operations in SQL

Union in SQL

```
select 1 union select 2 union select 3
```

Difference in SQL

```
(select 1 union select 2 union select 3) except select 2
```

Intersect in SQL

```
(select 1 union select 2 union select 3)
intersect (select 1 union select 2)
```

Bag vs. Set semantics in SQL

Attention

- Tables in SQL implement bag semantics
- `select` implements multisets/bag

Customer

ID	Name	ShippingAddress
1	Holger	180 Queens Gate
2	Sam	32 Vassar Street
3	Peter	180 Queens Gate

```
select ShippingAddress
from Customer
```

ShippingAddress
180 Queens Gate
32 Vassar Street
180 Queens Gate

```
select distinct
ShippingAddress from Customer
```

ShippingAddress
32 Vassar Street
180 Queens Gate

Bag vs. Set semantics in SQL

Other operators have other defaults

Operator	Default	Switching behaviour
Select	Bag	`select disctinct`
Union	Set	`union all`
Except	Set	`except all`
Intersect	Set	`intersect all`

Table aliasing

Tables can be renamed while querying

```
select ShippingAddress from Customer as c
```

Customer

ID	Name	ShippingAddress
1	Holger	180 Queens Gate
2	Sam	32 Vassar Street
3	Peter	180 Queens Gate

This is particularly important when crossing with itself

```
select c1.Name, c2.Name
from Customer as c1, Customer as c2
where c1.ShippingAddress = c2.ShippingAddress
and c1.Name <> c2.Name
```

Case Statements

Customer

ID	Name	ShippingAddress
1	Holger	180 Queens Gate
2	Sam	32 Vassar Street
3	Peter	180 Queens Gate

```
select id, case when Name = "Holger" then
"me" else "someone" end as who from Customer
```

ID	who
1	me
2	someone
3	someone

Null-Semantics in SQL

Quiz: What does a null-value mean

Unknown	We don't know the value of this attribute, may not even have one
Missing	The value exists but has not been entered into the system
Uncertain	The value exists but I have conflicting information about what it is
Invalid	The value that was entered in the system does not conform to the rules
Not an option	This column is here for technical reasons but is not to be set

Null-Semantics in SQL

Definition fuzzy

- SQL uses a three valued logic to process `WHERE` predicate
- Truth values are `TRUE`, `FALSE`, and `UNKNOWN`
- SQL standard vague, but handling of `NULL` is roughtly equivalent to "Unknown"

Truth values of SQL Formulae

Formula	Result
x =null	`UNKNOWN`
null=null	`UNKNOWN`
x IS NULL	`TRUE` if x has a null value, `FALSE` otherwise
x IS NOT NULL	`TRUE` if x does not have a null value, `FALSE` otherwise

Expression on null values produce null values

```
select c from (select null as c) as d where (1/c) is null
```

Aggregation

Aggregate functions - the five ansi SQL ones

- `Sum, Count, Min, Max, Avg`

Customer

ID	Name	ShippingAddress
1	Holger	180 Queens Gate
2	Sam	32 Vassar Street
3	Peter	180 Queens Gate

```
select count(*) as
count from Customer
```

count
3

```
select max(len(Name)) as
length from Customer
```

length
6

Null-Semantics in Aggregation

Min, Max, Avg, Sum

- Null values do not contribute
- They are simply ignored

Count is weird

`count(attribute)` null values are ignored

`count(*)` null values are included in count

Group by

Customer

ID	Name	ShippingAddress
1	Holger	180 Queens Gate
2	Sam	32 Vassar Street
3	Peter	180 Queens Gate

```
select count(*) as c
from Customer group by
ShippingAddress
```

c
2
1

```
select max(len(Name)) as
length from Customer group by
ShippingAddress
```

length
6
3

Null-Semantics in Group by

Rule

- Null values form a group of their own

Customer

ID	Name	ShippingAddress
1	Holger	180 Queens Gate
2	Sam	32 Vassar Street
3	Peter	180 Queens Gate
3	Daniel	NULL

```
select max(len(Name))
as l from Customer
group by
ShippingAddress
```

l
6
3
6

Quiz

Question

- What is the output of the following query

```
select count(*) from (select null as c
union all select null as c) as t group by c
```

Option 1	Option 2	Option 3	Option 4
1	2	NULL	empty

Quiz

Question

- What is the output of the following query

```
select v, count(c) from (
select 1 as v, cast(null as int) as c union all
select 2 as v, cast(null as int) as c) as t group by v
```

Option 1

1	0
2	0

Option 2

1	NULL
2	NULL

Option 3

1	1
2	1

Option 4

Empty

Having

Customer

ID	Name	ShippingAddress
1	Holger	180 Queens Gate
2	Sam	32 Vassar Street
3	Peter	180 Queens Gate

```
select count(*) as c from Customer group by
ShippingAddress having max(len(Name)) < 5
```

c
1

Order by

Customer

ID	Name	ShippingAddress
1	Holger	180 Queens Gate
2	Sam	32 Vassar Street
3	Peter	180 Queens Gate

```
select * from Customer
order by len(Name)
```

ID	Name	ShippingAddress
2	Sam	32 Vassar Street
3	Peter	180 Queens Gate
1	Holger	180 Queens Gate

top n

Customer

ID	Name	ShippingAddress
1	Holger	180 Queens Gate
2	Sam	32 Vassar Street
3	Peter	180 Queens Gate

```
select top 1 * from Customer
order by len(Name)
```

ID	Name	ShippingAddress
2	Sam	32 Vassar Street

Remember bag semantics!

- top n without order by is not properly defined

Things we're not talking about today

Operators

- `left join`
- `right join`
- `in`
- `some`
- `all`
- `exists`

The End