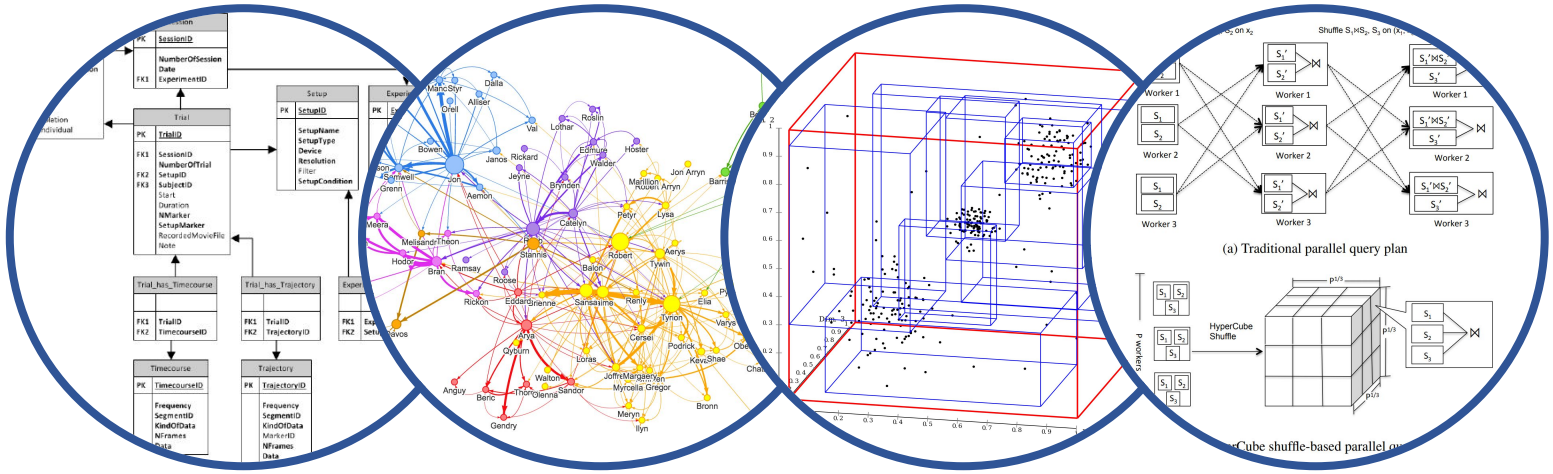


Introduction to Data Management

SQL++ … +

Alyssa Pittman
Based on slides by Jonathan Leang, Dan Suciu, et al

Paul G. Allen School of Computer Science and Engineering
University of Washington, Seattle

Announcements

- HW6 was due
  - Make sure you shut down your EMR clusters!!
- HW7 out today - Asterix

Recap: SQL++ Ideas

- SQL++ works over collections of data
  - Arrays […]
  - Multisets {{ … }}
- SQL++ is extremely similar to SQL
  - SQL++ for SQL Users by Don Chamberlin
  - For-each semantics
  - Main difference is nested data

Outline

- Unnesting walkthrough
  - Artificially nested data
- Nesting walkthrough

SQL++ Mini Demo

General Installation (Details in HW8 spec)

Download from:
https://asterixdb.apache.org/download.html

Start local cluster from:
<asterix root>/opt/local/bin/start-sample-cluster

Use web browser for interaction, default:
127.0.0.19001

Don't forget to stop cluster when you're done:
<asterix root>/opt/local/bin/stop-sample-cluster

SQL++ Mini Demo

General Usage:

Everything is running locally so make sure your computer doesn't die (advise against SELECT *)

Don't use attu, previous quarters people accidentally used other people's instance

Learn something! SQL++ may be a model for many future query languages.

Unnesting General Concept

- Semi-structured data often has nested values
- Unnesting is similar to the "flatmap" process
  - Bring nested data to the "top level"

Nested data: [[a], [a, b], [[a, b, c], d]]

Unnesting General Concept

- Semi-structured data often has nested values
- Unnesting is similar to the "flatmap" process
  - Bring nested data to the "top level"

Nested data: [[a], [a, b], [[a, b, c], d]]

...

Unnested: [a, a, b, a, b, c, d]

Unnesting Play-by-Play

- SQL
  1. Cross Products
  2. Selection filters
  3. …
- SQL++
  1. Unnesting
  2. Cross Products
  3. Selection filters
  4. …

Unnesting Play-by-Play

```
SELECT P.name, O.product
  FROM Person AS P, P.orders AS O
```

```
-- Dataset Person (simplified)
{{
   {
      "name": "Dan",
      "orders": [
         { "product": "Furby" }
      ]
   },
   {
      "name": "Alvin",
      "orders": [
         { "product": "Furby" },
         { "product": "Magic8" }
      ]
   },
   {
      "name": "Magda",
      "orders": []
   }
}}
```

Unnesting Play-by-Play

```
SELECT P.name, O.product
  FROM Person AS P UNNEST P.orders AS O
```

```
-- Dataset Person (simplified)
{{
   {
      "name": "Dan",
      "orders": [
         { "product": "Furby" }
      ]
   },
   {
      "name": "Alvin",
      "orders": [
         { "product": "Furby" },
         { "product": "Magic8" }
      ]
   },
   {
      "name": "Magda",
      "orders": []
   }
}}
```

Unnesting Play-by-Play

```
SELECT P.name, O.product
  FROM Person AS P UNNEST P.orders AS O
```

```
-- Dataset Person (simplified)
{{
   {
      "name": "Dan",
      "orders": [
         { "product": "Furby" }
      ]
   },
   {
      "name": "Alvin",
      "orders": [
         { "product": "Furby" },
         { "product": "Magic8" }
      ]
   },
   {
      "name": "Magda",
      "orders": []
   }
}}
```

Name	Orders
Dan	Product Furby
Alvin	Product Furby Magic8
Magda	Product

Unnesting Play-by-Play

```
SELECT P.name, O.product
  FROM Person AS P UNNEST P.orders AS O
```

Name	Orders
Dan	Product Furby
Alvin	Product Furby Magic8
Magda	Product

Unnesting Play-by-Play

```
SELECT P.name, O.product
  FROM Person AS P UNNEST P.orders AS O
```

Name	Orders
Dan	Product Furby
Alvin	Product Furby Magic8
Magda	Product

For each Person

Unnesting Play-by-Play

```
SELECT P.name, O.product
  FROM Person AS P UNNEST P.orders AS O
```

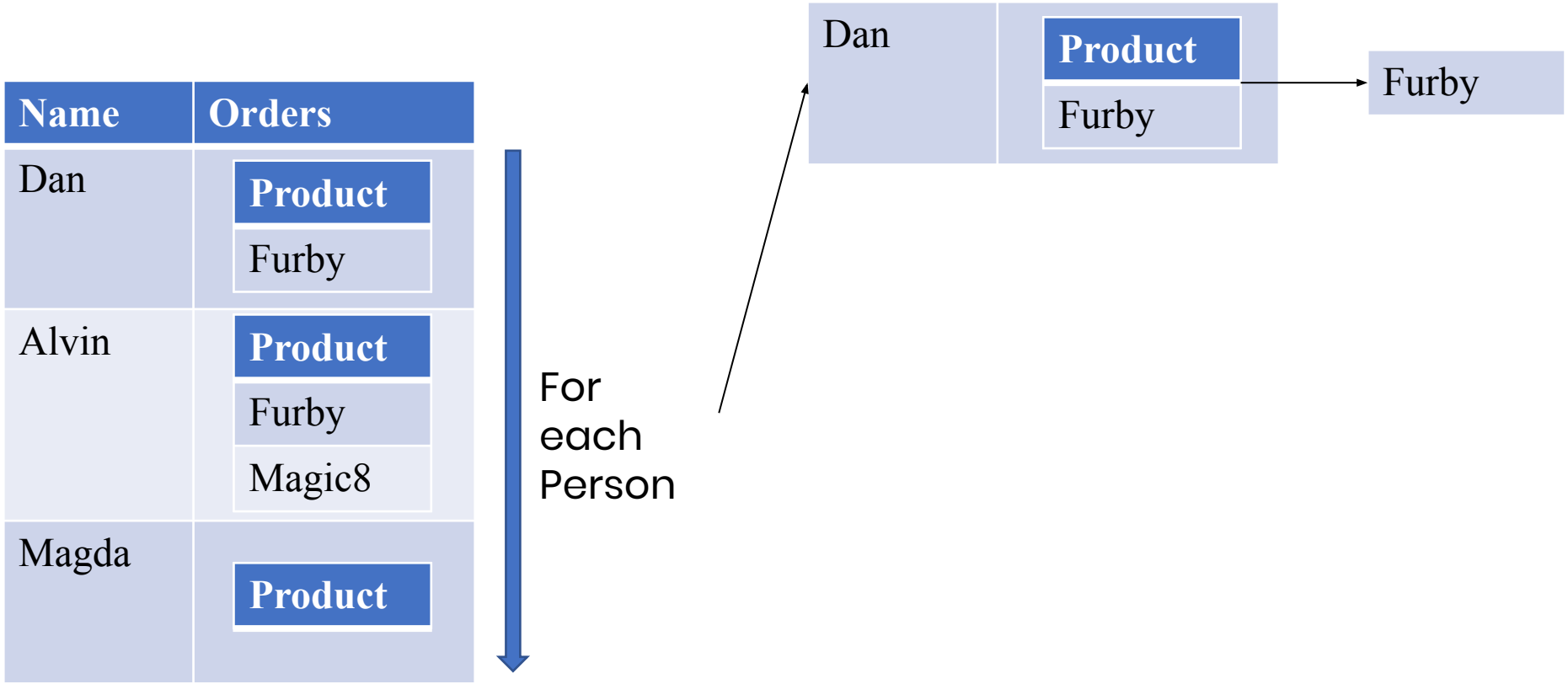
Name	Orders
Dan	Product Furby
Alvin	Product Furby Magic8
Magda	Product

For each Person

Dan	Product Furby

Unnesting Play-by-Play

```
SELECT P.name, O.product
  FROM Person AS P UNNEST P.orders AS O
```



Unnesting Play-by-Play

```
SELECT P.name, O.product
  FROM Person AS P UNNEST P.orders AS O
```

Name	Orders
Dan	Product Furby
Alvin	Product Furby Magic8
Magda	Product

For each Person

Dan	Product Furby

× Furby

Unnesting Play-by-Play

```
SELECT P.name, O.product
  FROM Person AS P UNNEST P.orders AS O
```

Name	Orders
Dan	Product Furby
Alvin	Product Furby Magic8
Magda	Product

For each Person

Dan	Product Furby	×	Furby

=

Dan	Product Furby	Furby

Unnesting Play-by-Play

```
SELECT P.name, O.product
  FROM Person AS P UNNEST P.orders AS O
```

Name	Orders
Dan	Product Furby
Alvin	Product Furby Magic8
Magda	Product

For each Person

Dan	Product Furby	× Furby

=

Dan	Product Furby	Furby

=

```
{
   "P": {
      "name": "Dan",
      "orders": [
         { "product": "Furby" }
      ]
   },
   "O": { "product": "Furby" }
}
```

Unnesting Play-by-Play

```
SELECT P.name, O.product
  FROM Person AS P UNNEST P.orders AS O
```

Name	Orders
Dan	Product Furby
Alvin	Product Furby Magic8
Magda	Product

For each Person

Alvin	Product Furby Magic8

Unnesting Play-by-Play

```
SELECT P.name, O.product
  FROM Person AS P UNNEST P.orders AS O
```

Name	Orders
Dan	Product Furby
Alvin	Product Furby Magic8
Magda	Product

For each Person

Alvin	Product
	Furby
	Magic8

Furby
Magic8

Unnesting Play-by-Play

```
SELECT P.name, O.product
  FROM Person AS P UNNEST P.orders AS O
```

Name	Orders
Dan	Product Furby
Alvin	Product Furby Magic8
Magda	Product

For each Person

Alvin	Product
	Furby
	Magic8

×

Furby
Magic8

Unnesting Play-by-Play

```
SELECT P.name, O.product
  FROM Person AS P UNNEST P.orders AS O
```

Name	Orders
Dan	Product Furby
Alvin	Product Furby Magic8
Magda	Product

For each Person

Alvin	Product Furby Magic8

×

Furby
Magic8

=

Alvin	Product Furby Magic8	Furby
Alvin	Product Furby Magic8	Magic8

Unnesting Play-by-Play

```
SELECT P.name, O.product
  FROM Person AS P UNNEST P.orders AS O
```

Name	Orders
Dan	Product Furby
Alvin	Product Furby Magic8
Magda	Product

For each Person

```
{
   "P": {
      "name": "Alvin",
      "orders": [
         { "product": "Furby" },
         { "product": "Magic8" }
      ]
   },
   "O": { "product": "Furby" }
},
{
   "P": {
      "name": "Alvin",
      "orders": [
         { "product": "Furby" },
         { "product": "Magic8" }
      ]
   },
   "O": { "product": "Magic8" }
}
```

Unnesting Play-by-Play

```
SELECT P.name, O.product
  FROM Person AS P UNNEST P.orders AS O
```

Name	Orders
Dan	Product Furby
Alvin	Product Furby Magic8
Magda	Product

For each Person

Magda	Product

Unnesting Play-by-Play

```
SELECT P.name, O.product
  FROM Person AS P UNNEST P.orders AS O
```

Name	Orders
Dan	Product Furby
Alvin	Product Furby Magic8
Magda	Product

For each Person

Magda	Product → ∅

Unnesting Play-by-Play

```
SELECT P.name, O.product
  FROM Person AS P UNNEST P.orders AS O
```

Name	Orders
Dan	Product Furby
Alvin	Product Furby Magic8
Magda	Product

For each Person

Magda	Product

× ∅

Unnesting Play-by-Play

```
SELECT P.name, O.product
  FROM Person AS P UNNEST P.orders AS O
```

Name	Orders
Dan	Product Furby
Alvin	Product Furby Magic8
Magda	Product

For each Person

Magda	Product

× ∅

=

∅

Unnesting Play-by-Play

```
SELECT P.name, O.product
  FROM Person AS P UNNEST P.orders AS O
```

Name	Orders
Dan	Product Furby
Alvin	Product Furby Magic8
Magda	Product

Dan	Product Furby	Furby
Alvin	Product Furby Magic8	Furby
Alvin	Product Furby Magic8	Magic8

Unnesting Play-by-Play

```
SELECT P.name, O.product
  FROM Person AS P UNNEST P.orders AS O
```

Dan	**Product** Furby	Furby
Alvin	**Product** Furby Magic8	Furby
Alvin	**Product** Furby Magic8	Magic8

For each

Unnesting Play-by-Play

```
SELECT P.name, O.product
  FROM Person AS P UNNEST P.orders AS O
```

Dan	**Product** Furby	Furby
Alvin	**Product** Furby Magic8	Furby
Alvin	**Product** Furby Magic8	Magic8

For each

```
{
   "P": {
      "name": "Dan",
      "orders": [
         { "product": "Furby" }
      ]
   },
   "O": { "product": "Furby" }
},
{
   "P": {
      "name": "Alvin",
      "orders": [
         { "product": "Furby" },
         { "product": "Magic8" }
      ]
   },
   "O": { "product": "Furby" }
},
{
   "P": {
      "name": "Alvin",
      "orders": [
         { "product": "Furby" },
         { "product": "Magic8" }
      ]
   },
   "O": { "product": "Magic8" }
}
```

Unnesting Play-by-Play

```
SELECT P.name, O.product
  FROM Person AS P UNNEST P.orders AS O
```

```
{
   "P": {
      "name": "Dan",
      "orders": [
         { "product": "Furby" }
      ]
   },
   "O": { "product": "Furby" }
},
{
   "P": {
      "name": "Alvin",
      "orders": [
         { "product": "Furby" },
         { "product": "Magic8" }
      ]
   },
   "O": { "product": "Furby" }
},
{
   "P": {
      "name": "Alvin",
      "orders": [
         { "product": "Furby" },
         { "product": "Magic8" }
      ]
   },
   "O": { "product": "Magic8" }
}
```

```
{
   "name": "Dan",
   "product": "Furby"
}
```

Unnesting Play-by-Play

```
SELECT P.name, O.product
  FROM Person AS P UNNEST P.orders AS O
```

```
{
   "P": {
      "name": "Dan",
      "orders": [
         { "product": "Furby" }
      ]
   },
   "O": { "product": "Furby" }
},
{
   "P": {
      "name": "Alvin",
      "orders": [
         { "product": "Furby" },
         { "product": "Magic8" }
      ]
   },
   "O": { "product": "Furby" }
},
{
   "P": {
      "name": "Alvin",
      "orders": [
         { "product": "Furby" },
         { "product": "Magic8" }
      ]
   },
   "O": { "product": "Magic8" }
}
```

```
{
   "name": "Dan",
   "product": "Furby"
}
```

Unnesting Play-by-Play

```
SELECT P.name, O.product
  FROM Person AS P UNNEST P.orders AS O
```

```
{
   "P": {
      "name": "Dan",
      "orders": [
         { "product": "Furby" }
      ]
   },
   "O": { "product": "Furby" }
},
{
   "P": {
      "name": "Alvin",
      "orders": [
         { "product": "Furby" },
         { "product": "Magic8" }
      ]
   },
   "O": { "product": "Furby" }
},
{
   "P": {
      "name": "Alvin",
      "orders": [
         { "product": "Furby" },
         { "product": "Magic8" }
      ]
   },
   "O": { "product": "Magic8" }
}
```

```
{
   "name": "Dan",
   "product": "Furby"
},
{
   "name": "Alvin",
   "product": "Furby"
},
{
   "name": "Alvin",
   "product": "Magic8"
}
```

Artificial Nesting

- Collections can be emulated by strings
  - Ex: Sentence □ Collection of words (tokens)
- Unnesting techniques still apply

Nested data: [“#NoSQL awesome”, “In Codd we trust”]

...

Unnested: [“#NoSQL”, “awesome”, “In”, “Codd”, “we”, “trust”]

Artificial Nesting

```
SELECT P.name, field
  FROM Person AS P, ???
```

```
{{
   {
      "name": "Dan",
      “desc": “DB_Theory"
   },
   {
      "name": "Alvin",
      “desc": “DB_PL"
   },
   {
      "name": "Magda",
      “desc": “DB_Systems"
   }
}}
```

Artificial Nesting

```
SELECT P.name, field
  FROM Person AS P, split(P.desc, "_") AS field
```

```
{{
   {
      "name": "Dan",
      "desc": "DB_Theory"
   },
   {
      "name": "Alvin",
      "desc": "DB_PL"
   },
   {
      "name": "Magda",
      "desc": "DB_Systems"
   }
}}
```

Artificial Nesting

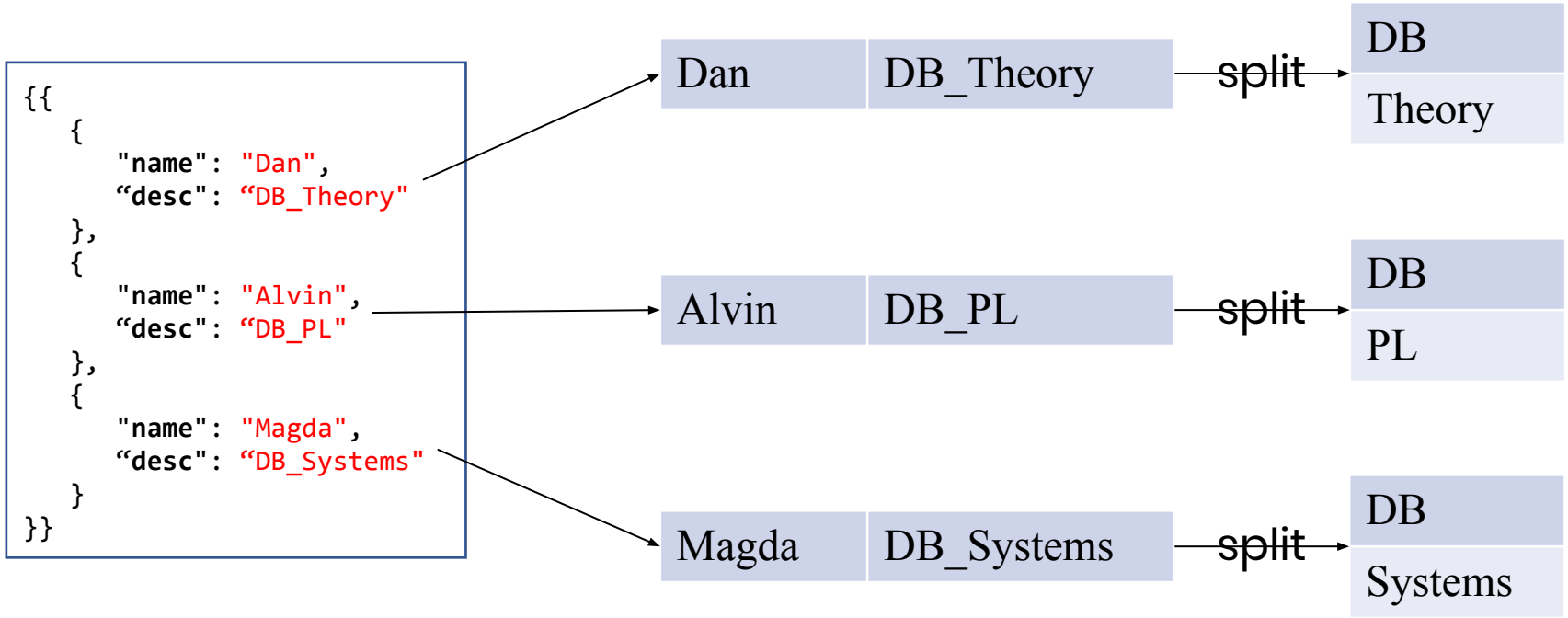
```
SELECT P.name, field
  FROM Person AS P, split(P.desc, "_") AS field
```

```
{{
   {
      "name": "Dan",
      "desc": "DB_Theory"
   },
   {
      "name": "Alvin",
      "desc": "DB_PL"
   },
   {
      "name": "Magda",
      "desc": "DB_Systems"
   }
}}
```

Dan	DB_Theory
Alvin	DB_PL
Magda	DB_Systems

Artificial Nesting

```
SELECT P.name, field
  FROM Person AS P, split(P.desc, "_") AS field
```



Artificial Nesting

```
SELECT P.name, field
  FROM Person AS P, split(P.desc, "_") AS field
```

```
{{
   {
      "name": "Dan",
      "desc": "DB_Theory"
   },
   {
      "name": "Alvin",
      "desc": "DB_PL"
   },
   {
      "name": "Magda",
      "desc": "DB_Systems"
   }
}}
```

Dan	DB_Theory	×	DB
			Theory
Alvin	DB_PL	×	DB
			PL
Magda	DB_Systems	×	DB
			Systems

Artificial Nesting

```
SELECT P.name, field
  FROM Person AS P, split(P.desc, "_") AS field
```

```
{{
   {
      "name": "Dan",
      "desc": "DB_Theory"
   },
   {
      "name": "Alvin",
      "desc": "DB_PL"
   },
   {
      "name": "Magda",
      "desc": "DB_Systems"
   }
}}
```

Dan	DB_Theory	DB
Dan	DB_Theory	Theory
Alvin	DB_PL	DB
Alvin	DB_PL	PL
Magda	DB_Systems	DB
Magda	DB_Systems	Systems

Artificial Nesting

```
SELECT P.name, field
  FROM Person AS P, split(P.desc, "_") AS field
```

Dan	DB_Theory	DB
Dan	DB_Theory	Theory
Alvin	DB_PL	DB
Alvin	DB_PL	PL
Magda	DB_Systems	DB
Magda	DB_Systems	Systems

Artificial Nesting

```
SELECT P.name, field
  FROM Person AS P, split(P.desc, "_") AS field
```

Dan	DB_Theory	DB
Dan	DB_Theory	Theory
Alvin	DB_PL	DB
Alvin	DB_PL	PL
Magda	DB_Systems	DB
Magda	DB_Systems	Systems

```
{
   ”P”: {
      "name": "Dan",
      “desc": “DB_Theory"
   },
   “field”: “DB"
},
{
   ”P”: {
      "name": "Dan",
      “desc": “DB_Theory"
   },
   “field”: “Theory"
},
{
   ”P”: {
      "name": "Alvin",
      “desc": “DB_PL"
   },
   “field”: “DB"
},
...
```

Artificial Nesting

```
SELECT P.name, field
  FROM Person AS P, split(P.desc, "_") AS field
```

```
{
   ”P”: {
      "name": "Dan",
      “desc": “DB_Theory"
   },
   “field”: “DB"
},
{
   ”P”: {
      "name": "Dan",
      “desc": “DB_Theory"
   },
   “field”: “Theory"
},
{
   ”P”: {
      "name": "Alvin",
      “desc": “DB_PL"
   },
   “field”: “DB"
},
...
```

Artificial Nesting

```
SELECT P.name, field
  FROM Person AS P, split(P.desc, "_") AS field
```

```
{
   ”P”: {
      "name": "Dan",
      “desc": “DB_Theory"
   },
   “field”: “DB"
},
{
   ”P”: {
      "name": "Dan",
      “desc": “DB_Theory"
   },
   “field”: “Theory"
},
{
   ”P”: {
      "name": "Alvin",
      “desc": “DB_PL"
   },
   “field”: “DB"
},
...
```

```
{
   "name": "Dan",
   “field”: “DB"
},
{
   "name": "Dan",
   “field”: “Theory"
},
{
   "name": "Alvin",
   “field”: “DB"
},
...
```

Unnesting and Joins

Find all conferences a person researches in

```
SELECT P.name, field, C.name
  FROM Person AS P, split(P.desc, "_") AS field,
       Conferences AS C
 WHERE C.topic = field
```

Unnesting and Joins

Find all conferences a person researches in

```
SELECT P.name, field, C.name
  FROM Person AS P, split(P.desc, "_") AS field,
       Conferences AS C
 WHERE C.topic = field
```

Unnest □ Join □ everything else

Unnesting and Joins

Find all conferences a person researches in

```
SELECT P.name, field, C.name
  FROM Person AS P, split(P.desc, "_") AS field,
       Conferences AS C
 WHERE C.topic = field
```

```
{
   "name": "Dan",
   "field": "DB"
},
{
   "name": "Dan",
   "field": "Theory"
},
{
   "name": "Alvin",
   "field": "DB"
},
...
```

Unnesting and Joins

Find all conferences a person researches in

```
SELECT P.name, field, C.name
  FROM Person AS P, split(P.desc, "_") AS field,
       Conferences AS C
 WHERE C.topic = field
```

```
{
   "name": "Dan",
   "field": "DB"
},
{
   "name": "Dan",
   "field": "Theory"
},
{
   "name": "Alvin",
   "field": "DB"
},
...
```

```
{
   "name": "SIGMOD",
   "topic": "DB"
},
{
   "name": "SIGACT",
   "topic": "Theory"
},
{
   "name": "SIGPLAN",
   "topic": "PL"
},
...
```

Unnesting and Joins

Find all conferences a person researches in

```
SELECT P.name, field, C.name
  FROM Person AS P, split(P.desc, "_") AS field,
       Conferences AS C
 WHERE C.topic = field
```

```
{
   "name": "Dan",
   "field": "DB"
},
{
   "name": "Dan",
   "field": "Theory"
},
{
   "name": "Alvin",
   "field": "DB"
},
...
```

$\bowtie_{field=topic}$

```
{
   "name": "SIGMOD",
   "topic": "DB"
},
{
   "name": "SIGACT",
   "topic": "Theory"
},
{
   "name": "SIGPLAN",
   "topic": "PL"
},
...
```

Unnesting and Joins

Find all conferences a person researches in

```
SELECT P.name, field, C.name
  FROM Person AS P, split(P.desc, "_") AS field,
       Conferences AS C
 WHERE C.topic = field
```

Name	Field
Dan	DB
Dan	Theory
Alvin	DB
…	…

$\bowtie_{field=topic}$

Name	Field
SIGMOD	DB
SIGACT	Theory
SIGPLAN	PL
…	…

Relational join!

Unnesting and Joins

Find all conferences a person researches in

```
SELECT P.name, field, C.name
  FROM Person AS P, split(P.desc, "_") AS field,
       Conferences AS C
 WHERE C.topic = field
```

Name	Field	Name
Dan	DB	SIGMOD
Dan	Theory	SIGACT
Alvin	DB	SIGMOD
…	…	…

Error!
Name collision

Unnesting and Joins

Find all conferences a person researches in

```
SELECT P.name AS pname, field, C.name AS cname
  FROM Person AS P, split(P.desc, "_") AS field,
       Conferences AS C
 WHERE C.topic = field
```

Pname	Field	Cname
Dan	DB	SIGMOD
Dan	Theory	SIGACT
Alvin	DB	SIGMOD
…	…	…

Unnesting and Joins

Find all conferences a person researches in

```
SELECT P.name AS pname, field, C.name AS cname
  FROM Person AS P, split(P.desc, "_") AS field,
       Conferences AS C
 WHERE C.topic = field
```

Pname	Field	Cname
Dan	DB	SIGMOD
Dan	Theory	SIGACT
Alvin	DB	SIGMOD
…	…	…

```
{
   "pname": "Dan",
   "cname": "SIGMOD",
   "topic": "DB"
},
{
   "pname": "Dan",
   "cname": "SIGACT",
   "topic": "Theory"
},
{
   "pname": "Alvin",
   "cname": "SIGMOD",
   "topic": "DB"
},
...
```

Nesting General Concept

- SQL++ is able to return semi-structured data
- Nesting is similar to the grouping process
  - Able to return collections of data for each group

Unnested data: [(x, a), (x, b), (y, a), (y, c), (z, b)]

Nesting General Concept

- SQL++ is able to return semi-structured data
- Nesting is similar to the grouping process
  - Able to return collections of data for each group

Unnested data: [(x, a), (x, b), (y, a), (y, c), (z, b)]

Nesting General Concept

- SQL++ is able to return semi-structured data
- Nesting is similar to the grouping process
  - Able to return collections of data for each group

Unnested data: [(x, a), (x, b), (y, a), (y, c), (z, b)]

Nested: [(x, [a, b]), (y, [a, c]), (z, [b])]

Nesting General Concept

- SQL++ is able to return semi-structured data
- Nesting is similar to the grouping process
  - Able to return collections of data for each group

Unnested data: [(x, a), (x, b), (y, a), (y, c), (z, b)]

Nested: [(x, [a, b]), (y, [a, c]), (z, [b])]

Important note:
GROUP BY and nesting are different! GROUP BY in SQL++ still obliterates ungrouped or unaggregated fields

Nesting Play-by-Play

For each conference, find all people who research the topic

```
SELECT I.cname, ???
  FROM Interests AS I
```

```
{
   ”pname”: “Dan",
   ”cname”: “SIGMOD",
   “topic”: “DB"
},
{
   ”pname”: “Dan",
   ”cname”: “SIGACT",
   “topic”: “Theory"
},
{
   ”pname”: “Alvin",
   ”cname”: “SIGMOD",
   “topic”: “DB"
}
```

Nesting Play-by-Play

For each conference, find all people who research the topic

```
SELECT I.cname, (SELECT I2.pname
                 FROM Interests AS I2
                 WHERE I2.topic = I.topic) AS people
  FROM Interests AS I
```

```
{
   ”pname”: “Dan",
   ”cname”: “SIGMOD",
   “topic”: “DB"
},
{
   ”pname”: “Dan",
   ”cname”: “SIGACT",
   “topic”: “Theory"
},
{
   ”pname”: “Alvin",
   ”cname”: “SIGMOD",
   “topic”: “DB"
}
```

Nesting Play-by-Play

For each conference, find all people who research the topic

```
SELECT I.cname, (SELECT I2.pname
                   FROM Interests AS I2
                  WHERE I2.topic = I.topic) AS people
  FROM Interests AS I
```

```
{
   "pname": "Dan",
   "cname": "SIGMOD",
   "topic": "DB"
},
{
   "pname": "Dan",
   "cname": "SIGACT",
   "topic": "Theory"
},
{
   "pname": "Alvin",
   "cname": "SIGMOD",
   "topic": "DB"
}
```

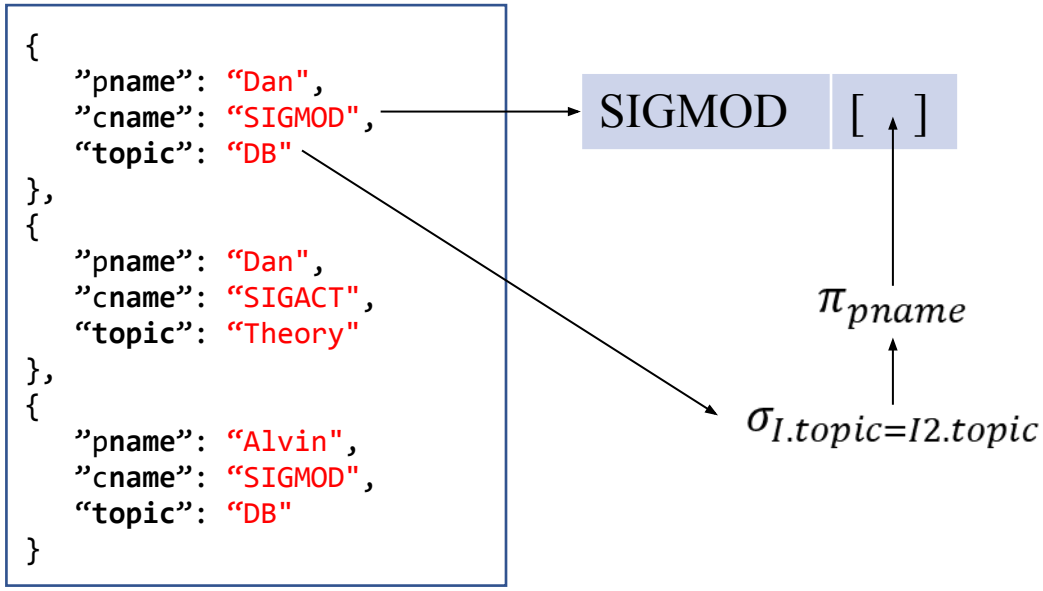
SIGMOD	[,]

```
SELECT I2.pname
  FROM Interests AS I2
 WHERE I2.topic = I.topic
```

Nesting Play-by-Play

For each conference, find all people who research the topic

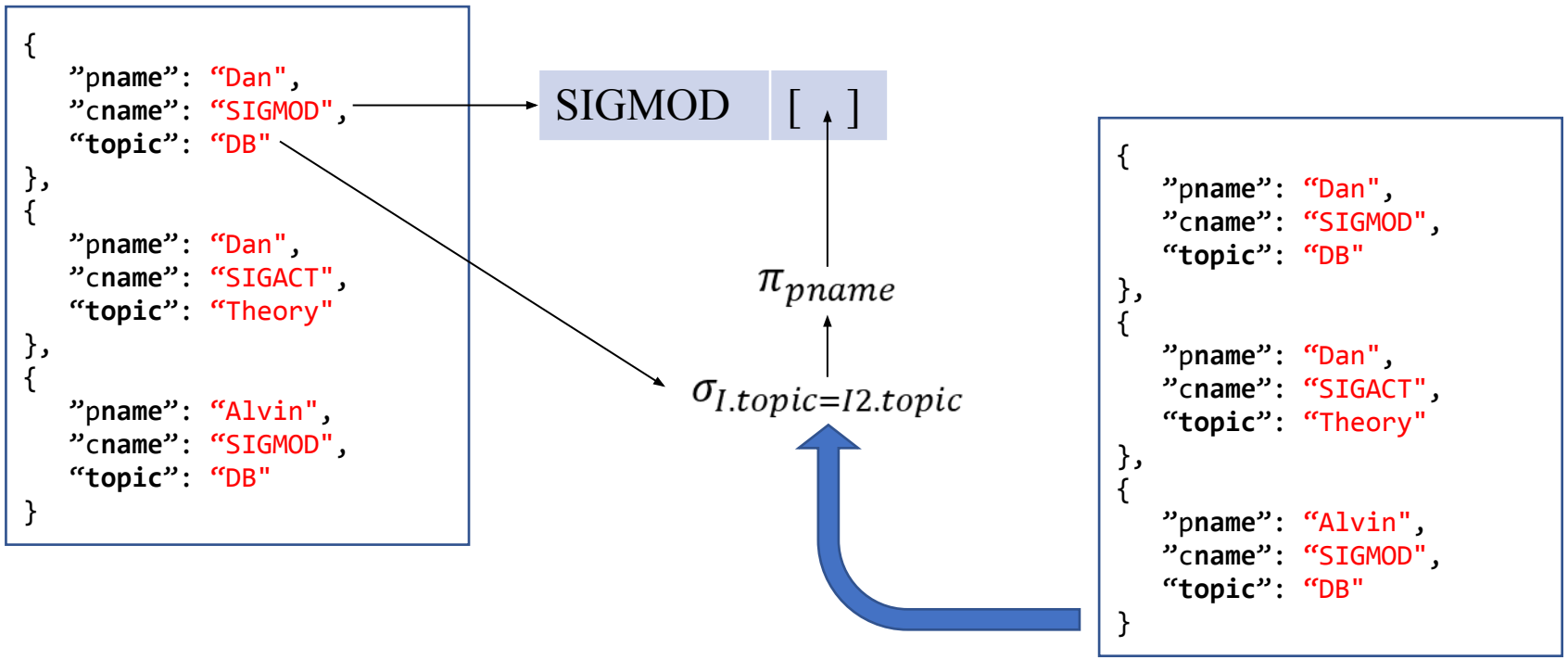
```
SELECT I.cname, (SELECT I2.pname
                   FROM Interests AS I2
                  WHERE I2.topic = I.topic) AS people
  FROM Interests AS I
```



Nesting Play-by-Play

For each conference, find all people who research the topic

```
SELECT I.cname, (SELECT I2.pname
                 FROM Interests AS I2
                 WHERE I2.topic = I.topic) AS people
  FROM Interests AS I
```



Nesting Play-by-Play

For each conference, find all people who research the topic

```
SELECT I.cname, (SELECT I2.pname
                   FROM Interests AS I2
                  WHERE I2.topic = I.topic) AS people
  FROM Interests AS I
```

```
{
   ”pname”: “Dan",
   ”cname”: “SIGMOD",
   “topic”: “DB"
},
{
   ”pname”: “Dan",
   ”cname”: “SIGACT",
   “topic”: “Theory"
},
{
   ”pname”: “Alvin",
   ”cname”: “SIGMOD",
   “topic”: “DB"
}
```

SIGMOD	[{“pname”:”Dan”}, {“pname”:”Alvin”}]

Nesting Play-by-Play

For each conference, find all people who research the topic

```
SELECT I.cname, (SELECT I2.pname
                   FROM Interests AS I2
                  WHERE I2.topic = I.topic) AS people
  FROM Interests AS I
```

```
{
   ”pname”: “Dan",
   ”cname”: “SIGMOD",
   “topic”: “DB"
},
{
   ”pname”: “Dan",
   ”cname”: “SIGACT",
   “topic”: “Theory"
},
{
   ”pname”: “Alvin",
   ”cname”: “SIGMOD",
   “topic”: “DB"
}
```

SIGMOD	[{“pname”:”Dan”}, {“pname”:”Alvin”}]
SIGACT	[{“pname”:”Dan”}]

Nesting Play-by-Play

For each conference, find all people who research the topic

```
SELECT I.cname, (SELECT I2.pname
                 FROM Interests AS I2
                 WHERE I2.topic = I.topic) AS people
  FROM Interests AS I
```

```
{
   "pname": "Dan",
   "cname": "SIGMOD",
   "topic": "DB"
},
{
   "pname": "Dan",
   "cname": "SIGACT",
   "topic": "Theory"
},
{
   "pname": "Alvin",
   "cname": "SIGMOD",
   "topic": "DB"
}
```

SIGMOD	[{"pname":"Dan"}, {"pname":"Alvin"}]
SIGACT	[{"pname":"Dan"}]
SIGMOD	[{"pname":"Dan"}, {"pname":"Alvin"}]

Duplicates!

Nesting Play-by-Play

For each conference, find all people who research the topic

```
SELECT DISTINCT I.cname, (SELECT I2.pname
                          FROM Interests AS I2
                         WHERE I2.topic = I.topic) AS people
  FROM Interests AS I
```

```
{
   "pname": "Dan",
   "cname": "SIGMOD",
   "topic": "DB"
},
{
   "pname": "Dan",
   "cname": "SIGACT",
   "topic": "Theory"
},
{
   "pname": "Alvin",
   "cname": "SIGMOD",
   "topic": "DB"
}
```

SIGMOD	[{"pname":"Dan"}, {"pname":"Alvin"}]
SIGACT	[{"pname":"Dan"}]
SIGMOD	~~[{"pname":"Dan"}, {"pname":"Alvin"}]~~

Nesting Play-by-Play

For each conference, find all people who research the topic

```
SELECT DISTINCT I.cname, (SELECT I2.pname
                          FROM Interests AS I2
                         WHERE I2.topic = I.topic) AS people
  FROM Interests AS I
```

```
{
   ”pname”: “Dan",
   ”cname”: “SIGMOD",
   “topic”: “DB"
},
{
   ”pname”: “Dan",
   ”cname”: “SIGACT",
   “topic”: “Theory"
},
{
   ”pname”: “Alvin",
   ”cname”: “SIGMOD",
   “topic”: “DB"
}
```

```
{
   ”cname”: “SIGMOD",
   ”people”: [
      { ”pname”: “Dan“ },
      { ”pname”: “Alvin“ }
   ]
},
{
   ”cname”: “SIGACT",
   ”people”: [
      { ”pname”: “Dan“ }
   ]
}
```

Nesting Play-by-Play

For each conference, find all people who research the topic

```
SELECT DISTINCT I.cname, (SELECT I2.pname
                          FROM Interests AS I2
                         WHERE I2.topic = I.topic) AS people
  FROM Interests AS I
```

```
{
   ”pname”: “Dan",
   ”cname”: “SIGMOD",
   “topic”: “DB"
},
{
   ”pname”: “Dan",
   ”cname”: “SIGACT",
   “topic”: “Theory"
},
{
   ”pname”: “Alvin",
   ”cname”: “SIGMOD",
   “topic”: “DB"
}
```

```
{
   ”cname”: “SIGMOD",
   ”people”: [
      { ”pname”: “Dan“ },
      { ”pname”: “Alvin“ }
   ]
},
{
   ”cname”: “SIGACT",
   ”people”: [
      { ”pname”: “Dan“ }
   ]
}
```

Objects look kinda ugly…

Nesting Play-by-Play

For each conference, find all people who research the topic

```
SELECT DISTINCT I.cname, (SELECT VALUE I2.pname
                          FROM Interests AS I2
                         WHERE I2.topic = I.topic) AS people
  FROM Interests AS I
```

```
{
   ”pname”: “Dan",
   ”cname”: “SIGMOD",
   “topic”: “DB"
},
{
   ”pname”: “Dan",
   ”cname”: “SIGACT",
   “topic”: “Theory"
},
{
   ”pname”: “Alvin",
   ”cname”: “SIGMOD",
   “topic”: “DB"
}
```

VALUE keyword extracts a the raw value from the specified field

```
{
   ”cname”: “SIGMOD",
   ”people”: [ “Dan“ , “Alvin“ ]
},
{
   ”cname”: “SIGACT",
   ”people”: [ “Dan“ ]
}
```

Nesting Play-by-Play

For each conference, find all people who research the topic

```
SELECT DISTINCT I.cname, people
  FROM Interests AS I
   LET people = (SELECT VALUE I2.pname
                   FROM Interests AS I2
                  WHERE I2.topic = I.topic)
```

Very readable!

```
{
   ”pname”: “Dan",
   ”cname”: “SIGMOD",
   “topic”: “DB"
},
{
   ”pname”: “Dan",
   ”cname”: “SIGACT",
   “topic”: “Theory"
},
{
   ”pname”: “Alvin",
   ”cname”: “SIGMOD",
   “topic”: “DB"
}
```

```
{
   ”cname”: “SIGMOD",
   ”people”: [ “Dan“ , “Alvin“ ]
},
{
   ”cname”: “SIGACT",
   ”people”: [ “Dan“ ]
}
```

SQL++ Aggregation

Lets you decide how to deal with 3-valued logic!

NULL considered (STRICT_SUM : if one value is null, return null)

NULL ignored (same as vanilla SQL)

Function	NULL	MISSING	Empty Collection
STRICT_COUNT	counted	counted	0
STRICT_SUM	returns NULL	returns NULL	returns NULL
STRICT_MAX	returns NULL	returns NULL	returns NULL
STRICT_MIN	returns NULL	returns NULL	returns NULL
STRICT_AVG	returns NULL	returns NULL	returns NULL
ARRAY_COUNT	not counted	not counted	0
ARRAY_SUM	ignores NULL	ignores NULL	returns NULL
ARRAY_MAX	ignores NULL	ignores NULL	returns NULL
ARRAY_MIN	ignores NULL	ignores NULL	returns NULL
ARRAY_AVG	ignores NULL	ignores NULL	returns NULL

SQL++ Aggregation

Lets you decide how to deal with 3-valued logic!

NULL considered (STRICT_SUM : if one value is null, return null)

NULL ignored (same as vanilla SQL)

Function	NULL	MISSING	Empty Collection
STRICT_COUNT	counted	counted	0
STRICT_SUM	returns NULL	returns NULL	returns NULL
STRICT_MAX	returns		urns NULL
STRICT_MIN	returns		urns NULL
STRICT_AVG	ret		urns NULL
ARRAY_COUNT	not counted	not counted	0
ARRAY_SUM	ignores NULL	ignores NULL	returns NULL
ARRAY_MAX	ignores NULL	ignores NULL	returns NULL
ARRAY_MIN	ignores NULL	ignores NULL	returns NULL
ARRAY_AVG	ignores NULL	ignores NULL	returns NULL

Bare COUNT, SUM, etc are aliased to ARRAY_COUNT, etc…