The Age of Data Conversation: Talk to Your Relational Data

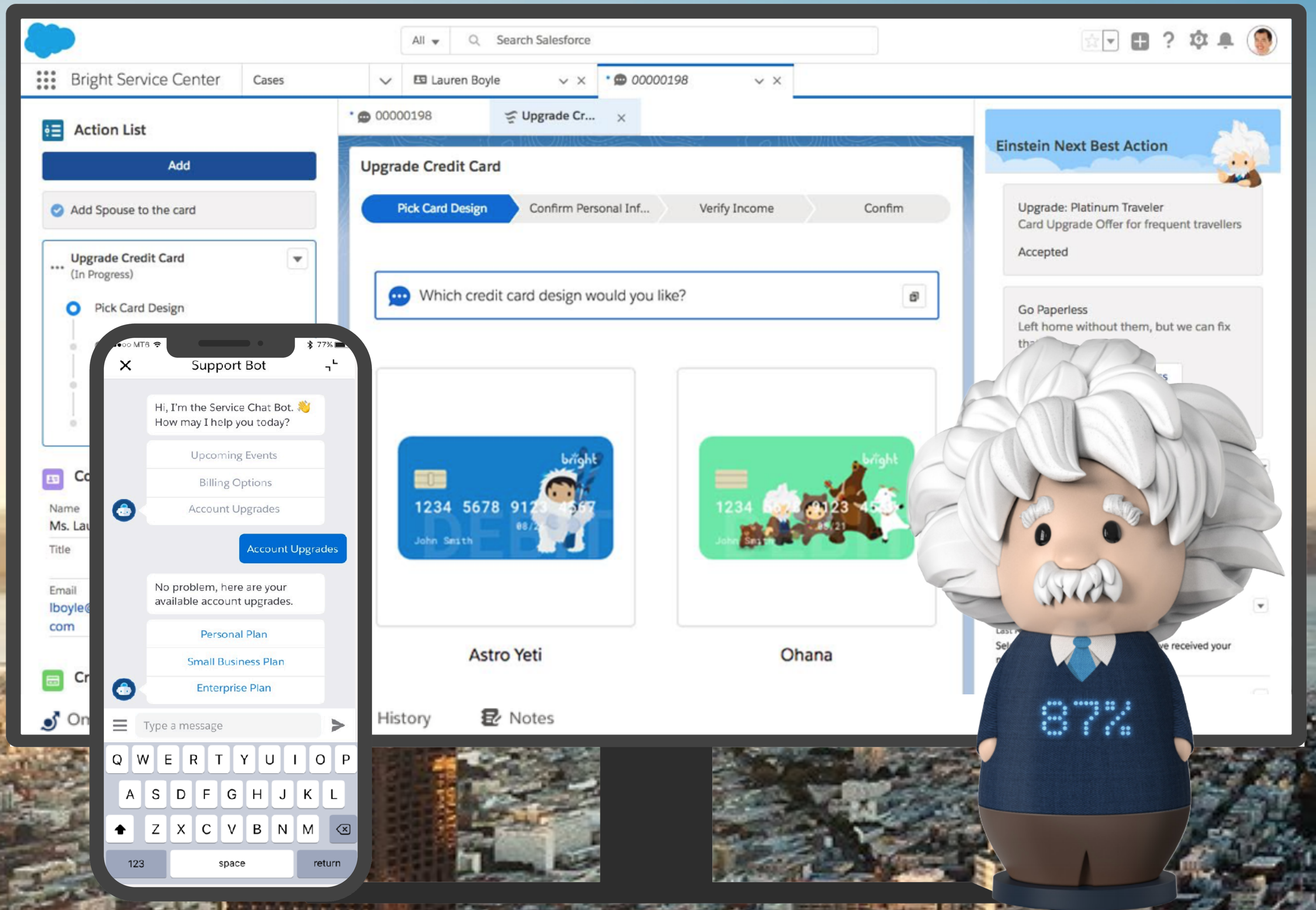
Victoria Lin

Senior Research Scientist, Salesforce AI Research

Dec 1, 2020

Joint work w/ Tao Yu[a], Chien-Sheng Wu, Rui Zhang[b], Bailin Wang[c], Karthik Radhakrishnan[d], Arvind Srikantan, Dragomir Radev[a], Richard Socher and Caiming Xiong

a - Yale University b - Penn State University
c - University of Edinburgh d - Carnegie Mellon University



Which credit card design would you like?

Upgrade: Platinum Traveler
Card Upgrade Offer for frequent travellers
Accepted

Go Paperless
Left home without them, but we can fix tha

Hi, I'm the Service Chat Bot. 👋
How may I help you today?

No problem, here are your available account upgrades.

We just acquired
Slack today!

Recap: Semantic Parsing

- **General definition:** natural language → formal meaning representations

NL: John likes fruits

LF: $\forall x\ \textsc{Fruit}(x) \implies \textsc{Likes}(x, \textsc{John})$

Recap: Semantic Parsing

- **General definition:** natural language → formal meaning representations

NL: John likes fruits

LF: $\forall x\ \textsc{Fruit}(x) \implies \textsc{Likes}(x, \textsc{John})$ (lambda expressions, CCG, AMR etc.)

Recap: Semantic Parsing

- **General definition:** natural language → formal meaning representations

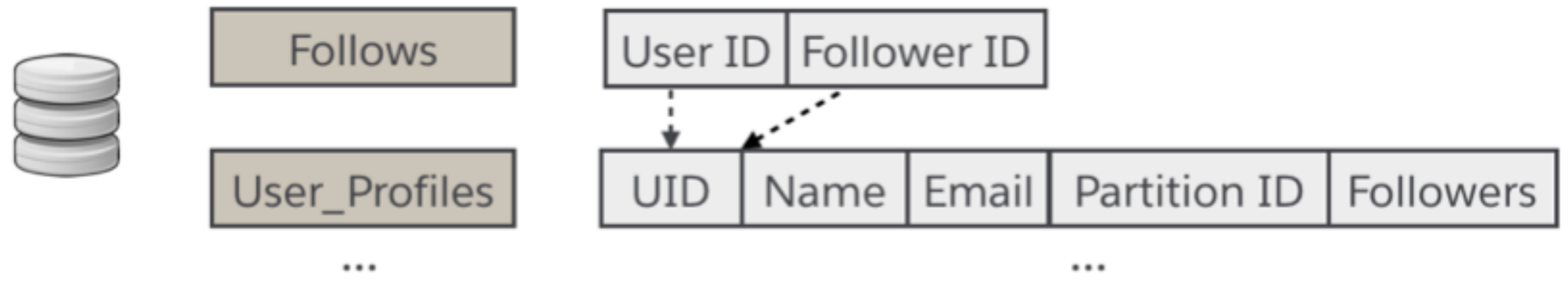
  NL: John likes fruits

  LF: $\forall x\ \textsc{Fruit}(x) \Longrightarrow \textsc{Likes}(x, \textsc{John})$ (lambda expressions, CCG, AMR etc.)

- **NL2Code:** natural language → high-level programming languages

  NL: List all users

  LF: `SELECT Name FROM User_Profiles`



Intelligent

Personal

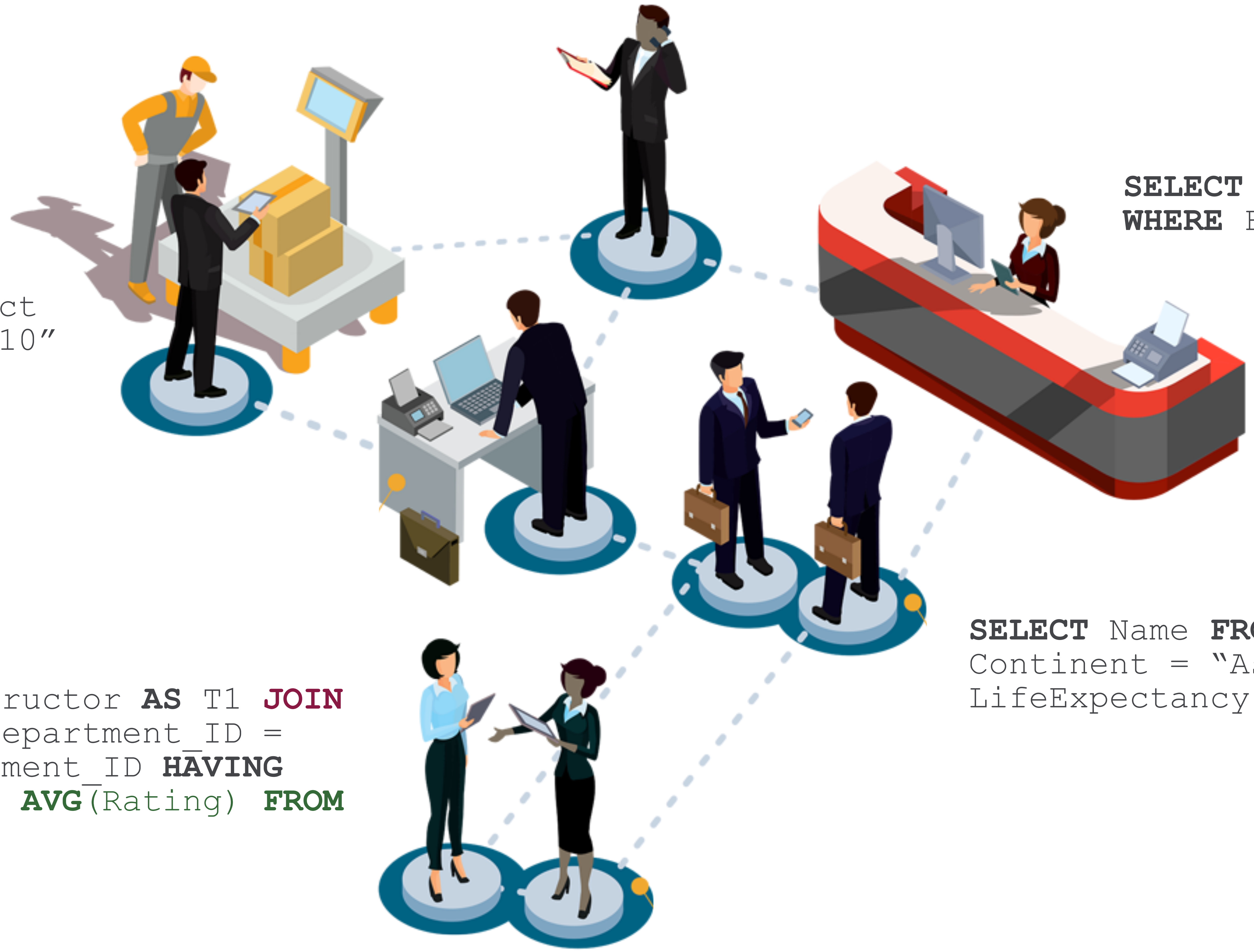
Conversational

Tables and **relational databases** are dominant data structures that powers the downstream AI applications.

The **Internet** contains billions of public tables. And more exist on **corporate intranets** and **personal devices**.

Natural Language Interface to Databases

Traditionally, database information is accessed using **structured query language (SQL)**.



```
SELECT Quantity FROM Product
WHERE Name = “Hoverboard x10”
```

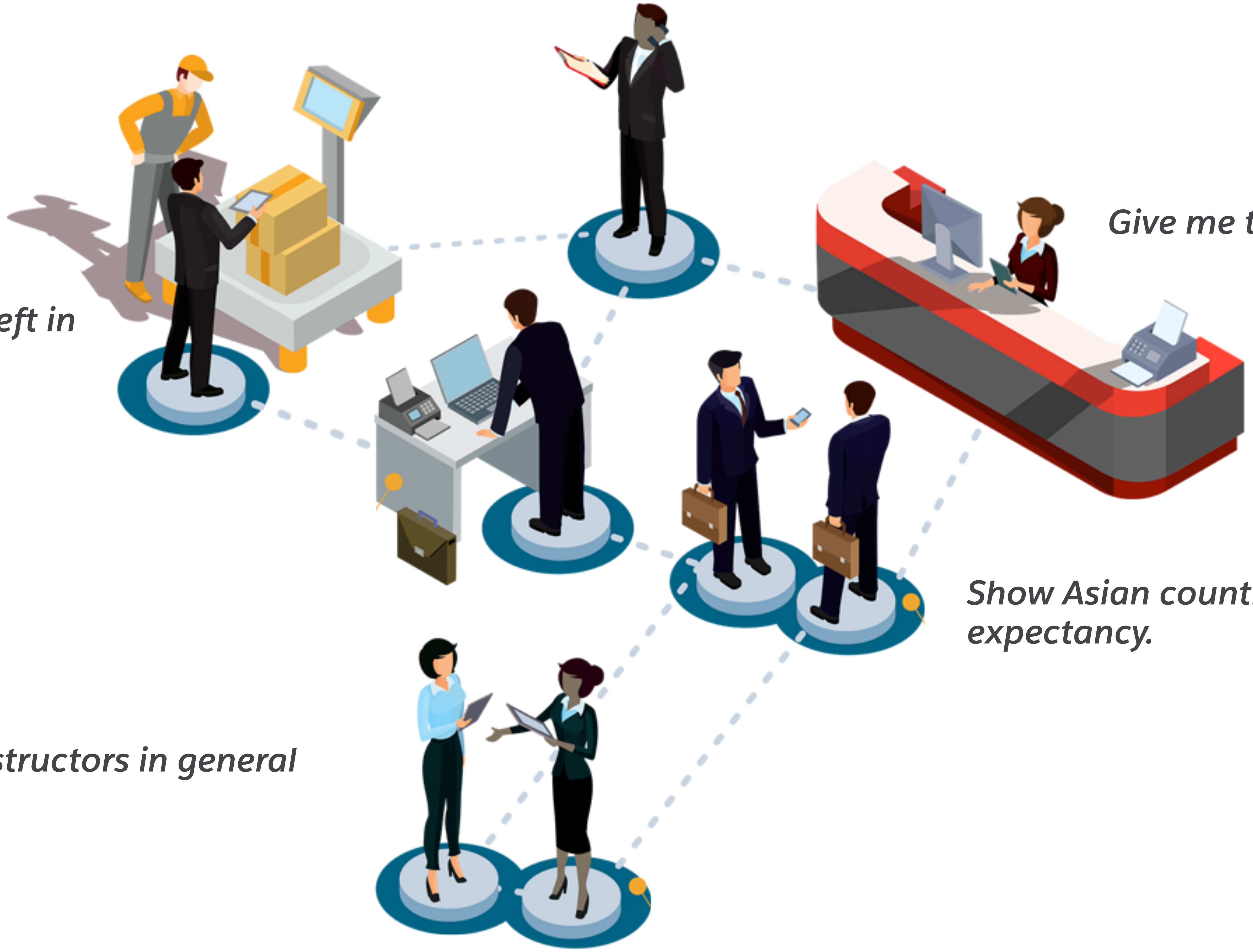
```
SELECT Arriving_Time FROM Flights
WHERE Flight_Number = “CZ327”
```

```
SELECT T2.name FROM Instructor AS T1 JOIN
Department AS T2 ON T1.Department_ID =
T2.ID GROUP BY T1.Department_ID HAVING
AVG(T1.Rating) > (SELECT AVG(Rating) FROM
Instructor)
```

```
SELECT Name FROM Country WHERE
Continent = “Asia” ORDER BY
LifeExpectancy LIMIT 1
```

Natural Language Interface to Databases

Our goal is to learn semantic parsers over **tables and databases** that maps natural language utterances to **executable** database queries**.**



Give me the arriving time of "CZ327".

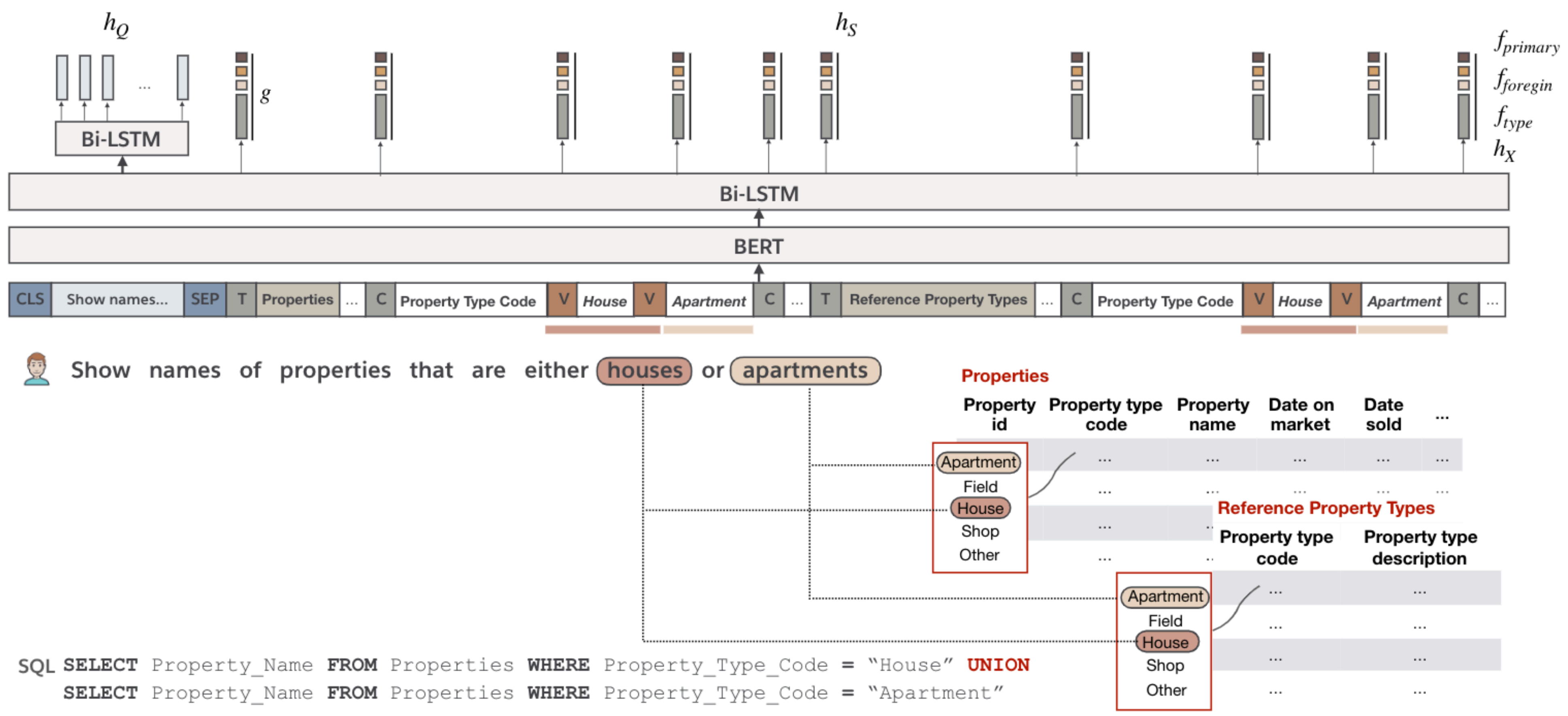
How many "Hoverboard x10" are left in stock?

Show Asian countries ordered by life expectancy.

Which departments have instructors in general rated above average?

I. Content-Aware Textual-Tabular Encodings for Table Semantic Parsing (TSP)

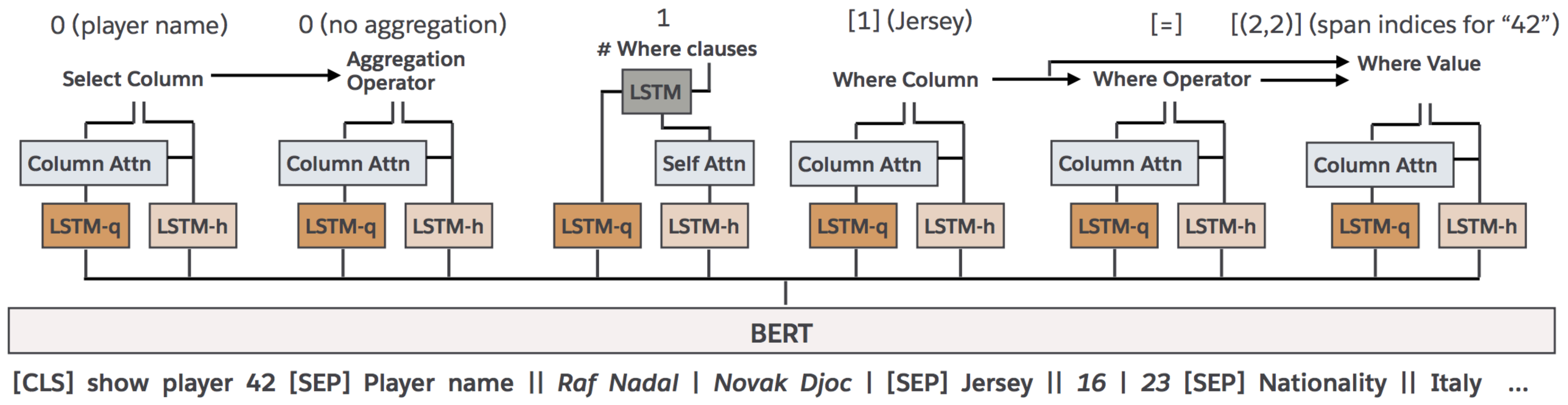
Bridging Textual and Tabular Data for Cross-Domain Text-to-SQL Semantic Parsing. Lin et al. 2020.



Show names of properties that are either houses or apartments

```
SQL SELECT Property_Name FROM Properties WHERE Property_Type_Code = "House" UNION
    SELECT Property_Name FROM Properties WHERE Property_Type_Code = "Apartment"
```

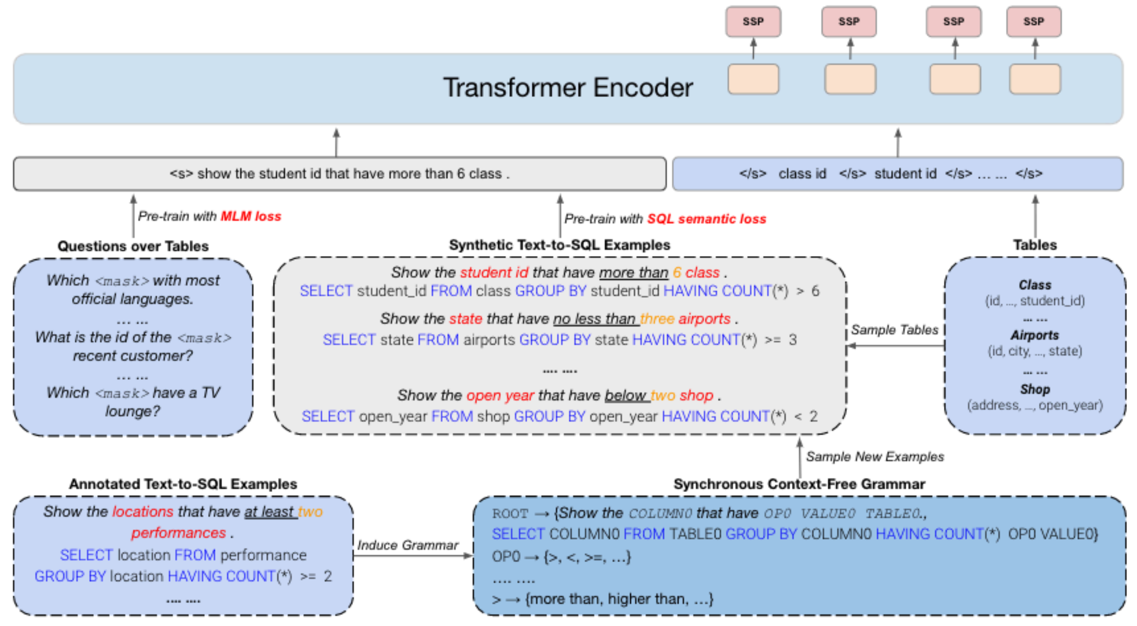
ColloQL: Robust Cross-Domain Text-to-SQL over Search Queries. Radhakrishnan et al. 2020.



[CLS] show player 42 [SEP] Player name || *Raf Nadal* | *Novak Djoc* | [SEP] Jersey || *16* | *23* [SEP] Nationality || Italy ...

II. Pre-training Textual-Tabular Representations with Semantic Scaffolds

GraPPa: Grammar-Augmented Pre-training for Table Semantic Parsing. Yu et al. 2020.


SSP
SSP
SSP
SSP
Transformer Encoder
<s> show the student id that have more than 6 class .
</s> class id </s> student id </s> ... ... </s>
Pre-train with MLM loss
Pre-train with SQL semantic loss
Questions over Tables
Synthetic Text-to-SQL Examples
Tables
Sample Tables
Class
(id, ..., student_id)
Airports
(id, city, ..., state)
Shop
(address, ..., open_year)
Sample New Examples
Annotated Text-to-SQL Examples
Induce Grammar
Synchronous Context-Free Grammar

Which <mask> with most official languages.
... ...
What is the id of the <mask> recent customer?
... ...
Which <mask> have a TV lounge?

Show the student id that have more than 6 class .
SELECT student_id FROM class GROUP BY student_id HAVING COUNT(*) > 6
Show the state that have no less than three airports .
SELECT state FROM airports GROUP BY state HAVING COUNT(*) >= 3
....
Show the open year that have below two shop .
SELECT open_year FROM shop GROUP BY open_year HAVING COUNT(*) < 2

Show the locations that have at least two performances .
SELECT location FROM performance GROUP BY location HAVING COUNT(*) >= 2
....

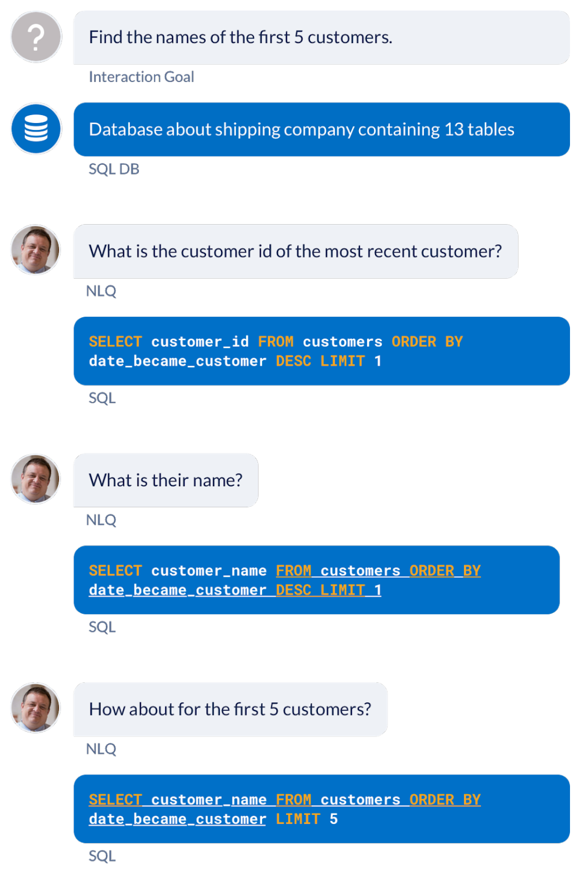
ROOT → {*Show the* COLUMN0 *that have* OP0 VALUE0 TABLE0.,
SELECT COLUMN0 FROM TABLE0 GROUP BY COLUMN0 HAVING COUNT(*) OP0 VALUE0}
OP0 → {>, <, >=, ...}
....
> → {more than, higher than, ...}

III. Conversational Table Semantic Parsing

SParC: Cross-Domain Semantic Parsing in Context. Yu et al. 2019.

Editing-Based SQL Query Generation for Cross-Domain Context-Dependent Questions. Zhang et al. 2019.

CoSQL: A Conversational Text-to-SQL Challenge Towards Cross-Domain Natural Language Interfaces to Databases. Yu et al. 2019.



Find the names of the first 5 customers.
Interaction Goal

Database about shipping company containing 13 tables
SQL DB

What is the customer id of the most recent customer?
NLQ

```
SELECT customer_id FROM customers ORDER BY
date_became_customer DESC LIMIT 1
```
SQL

What is their name?
NLQ

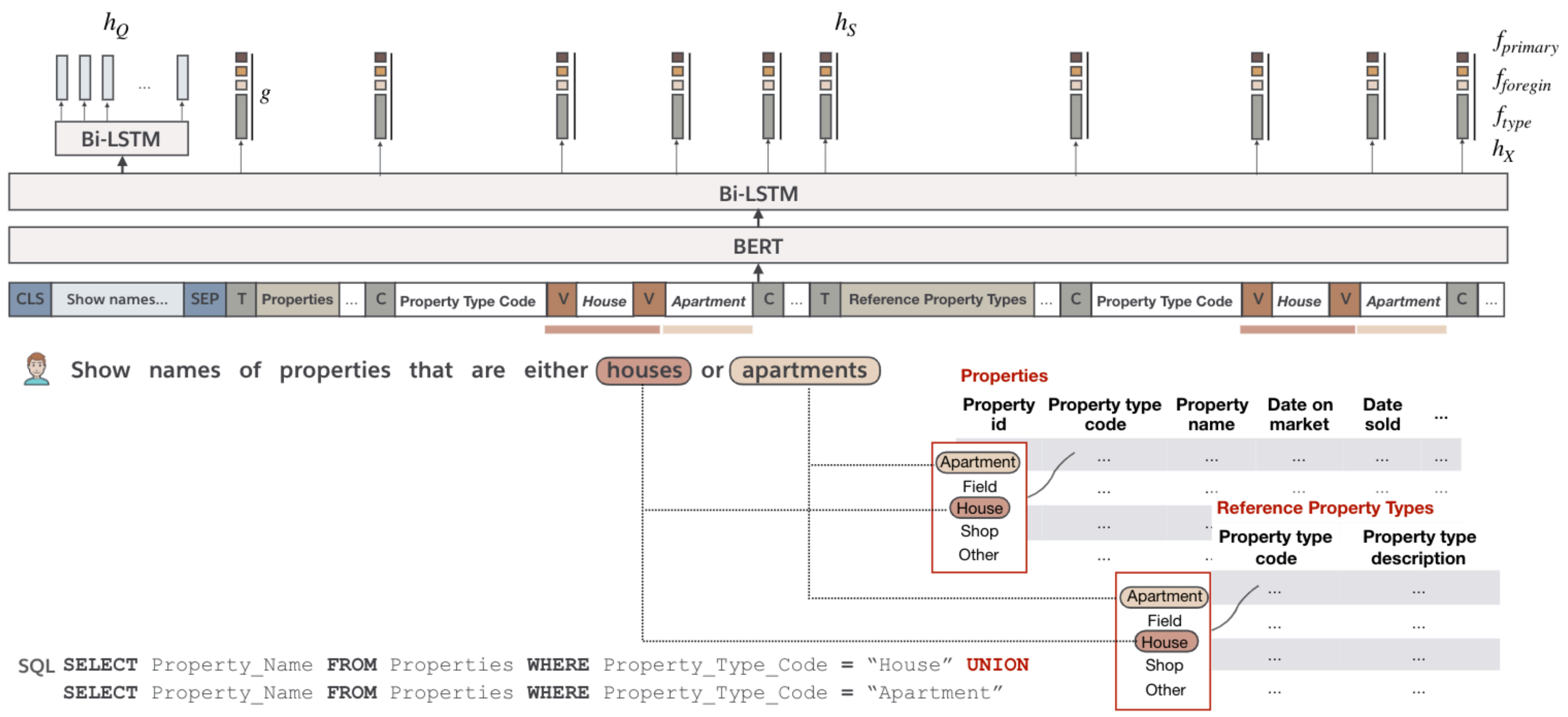
```
SELECT customer_name FROM customers ORDER BY
date_became_customer DESC LIMIT 1
```
SQL

How about for the first 5 customers?
NLQ

```
SELECT customer_name FROM customers ORDER BY
date_became_customer LIMIT 5
```
SQL

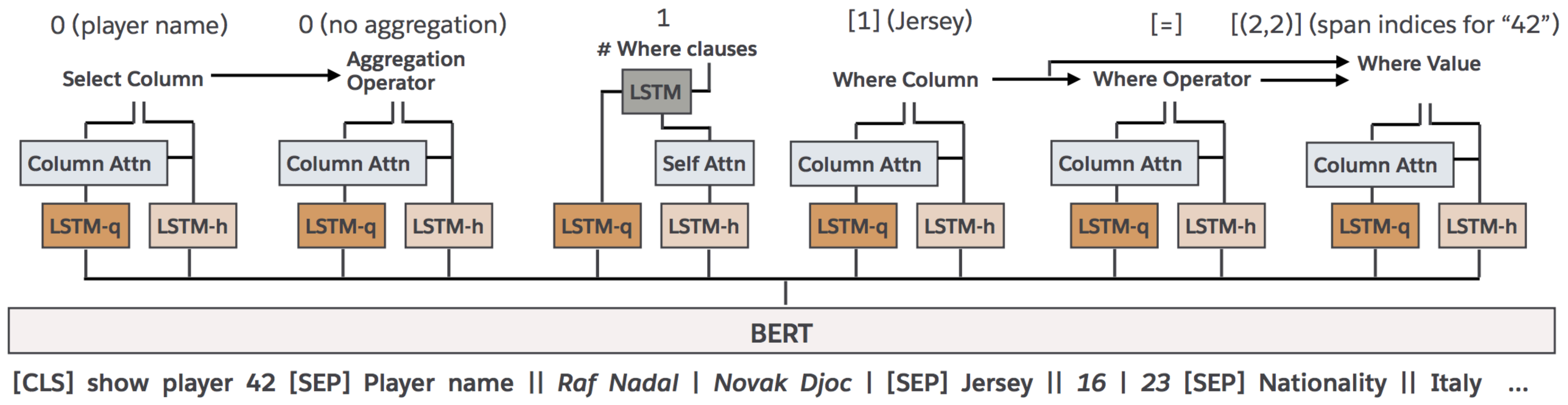
I. Content-Aware Textual-Tabular Encodings for Table Semantic Parsing (TSP)

Bridging Textual and Tabular Data for Cross-Domain Text-to-SQL Semantic Parsing. Lin et al. 2020.



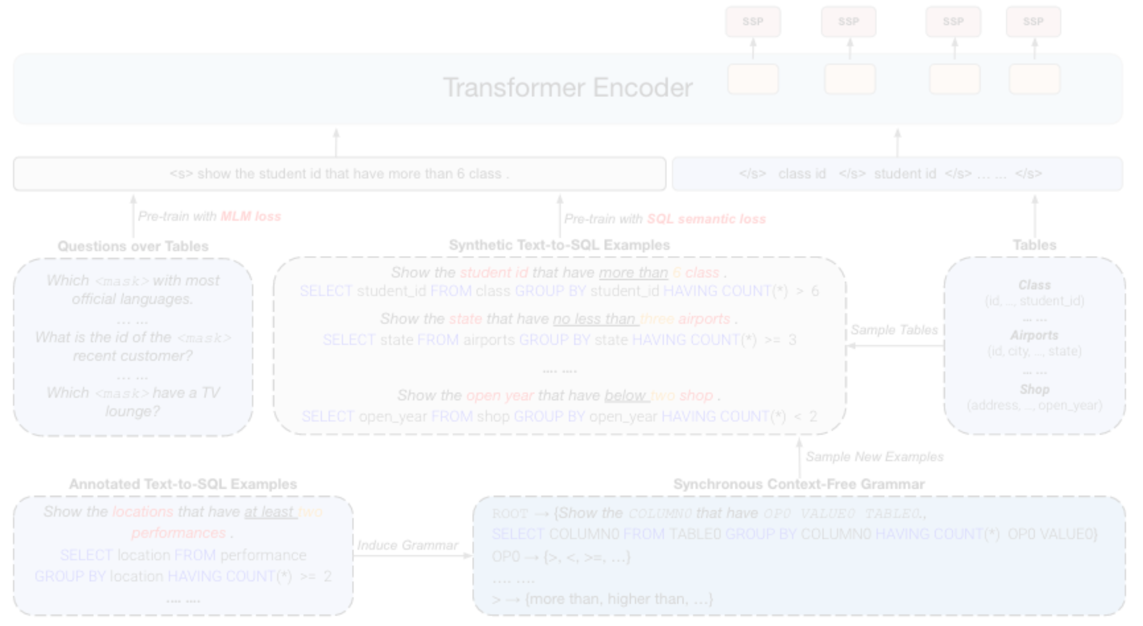
```
SQL SELECT Property_Name FROM Properties WHERE Property_Type_Code = "House" UNION
    SELECT Property_Name FROM Properties WHERE Property_Type_Code = "Apartment"
```

ColloQL: Robust Cross-Domain Text-to-SQL over Search Queries. Radhakrishnan et al. 2020.



II. Pre-training Textual-Tabular Representations with Semantic Scaffolds

GraPPa: Grammar-Augmented Pre-training for Table Semantic Parsing. Yu et al. 2020.

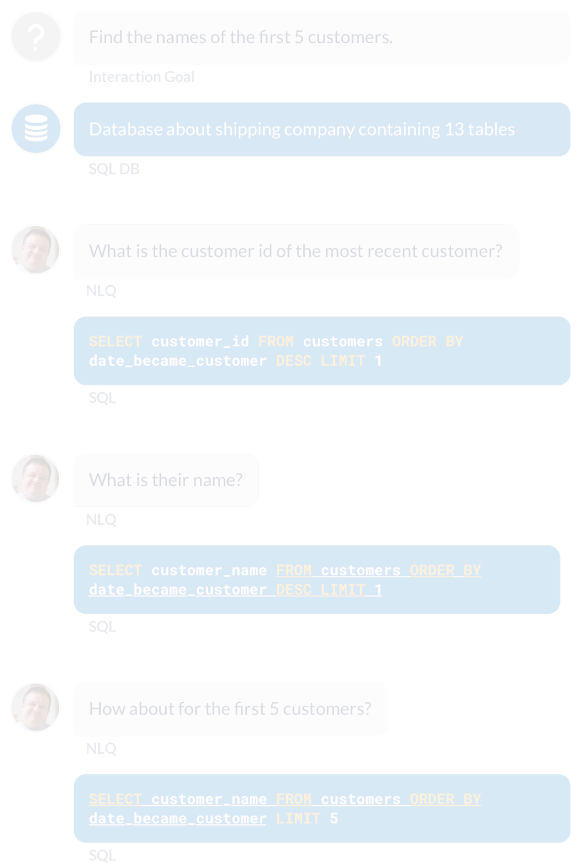

SSP
SSP
SSP
SSP
Transformer Encoder
<s> show the student id that have more than 6 class .
</s> class id </s> student id </s> ... ... </s>
Pre-train with MLM loss
Pre-train with SQL semantic loss
Questions over Tables
Which <mask> with most official languages.
... ...
What is the id of the <mask> recent customer?
... ...
Which <mask> have a TV lounge?
Synthetic Text-to-SQL Examples
Show the student id that have more than 6 class .
SELECT student_id FROM class GROUP BY student_id HAVING COUNT(*) > 6
Show the state that have no less than three airports .
SELECT state FROM airports GROUP BY state HAVING COUNT(*) >= 3
....
Show the open year that have below two shop .
SELECT open_year FROM shop GROUP BY open_year HAVING COUNT(*) < 2
Tables
Class
(id, ..., student_id)
... ...
Airports
(id, city, ..., state)
... ...
Shop
(address, ..., open_year)
Sample Tables
Sample New Examples
Annotated Text-to-SQL Examples
Show the locations that have at least two performances .
SELECT location FROM performance GROUP BY location HAVING COUNT(*) >= 2
....
Induce Grammar
Synchronous Context-Free Grammar
ROOT → {Show the COLUMN0 that have OP0 VALUE0 TABLE0.,
SELECT COLUMN0 FROM TABLE0 GROUP BY COLUMN0 HAVING COUNT(*) OP0 VALUE0}
OP0 → {>, <, >=, ...}
....
> → {more than, higher than, ...}

III. Conversational Table Semantic Parsing

SParC: Cross-Domain Semantic Parsing in Context. Yu et al. 2019.

Editing-Based SQL Query Generation for Cross-Domain Context-Dependent Questions. Zhang et al. 2019.

CoSQL: A Conversational Text-to-SQL Challenge Towards Cross-Domain Natural Language Interfaces to Databases. Yu et al. 2019.



Find the names of the first 5 customers.

Interaction Goal

Database about shipping company containing 13 tables

SQL DB

What is the customer id of the most recent customer?

NLQ

```
SELECT customer_id FROM customers ORDER BY
date_became_customer DESC LIMIT 1
```

SQL

What is their name?

NLQ

```
SELECT customer_name FROM customers ORDER BY
date_became_customer DESC LIMIT 1
```

SQL

How about for the first 5 customers?

NLQ

```
SELECT customer_name FROM customers ORDER BY
date_became_customer LIMIT 5
```

SQL

Table Semantic Parsing: A Brief History

ATIS Corpus collection

Hemphill et al. 1990
Dahl et al. 1994

Learning logical-form based semantic parsers for NLIDBs

Zelle and Mooney 1996
Popescu et al. 2003
Zettlemoyer and Collins 2005

Neural networks widely adopted in NLP

Sutskever et al. 2014
Bahdanau et al. 2015

Seq2Seq-style neural semantic parsing

Dong and Lapata 2016

WikiSQL: a large-scale, cross-domain text-to-SQL corpora

Zhong et al. 2017

TypeSQL, column attention, sketch-based, execution guided, RL, meta-learning

Xu et al. 2017
Dong and Lapata 2018
Wang et al. 2018
Yu et al. 2018a

Spider: expert-annotated, large-scale, cross-domain, complex

Yu et al. 2018b

LM-based pre-training: BERT

Devlin et al. 2018

Table-Aware BERT Encoder, surpassed human-performance on WikiSQL

Hwang et al. 2019

Syntax-guided, GNN, schema linking, SemQL

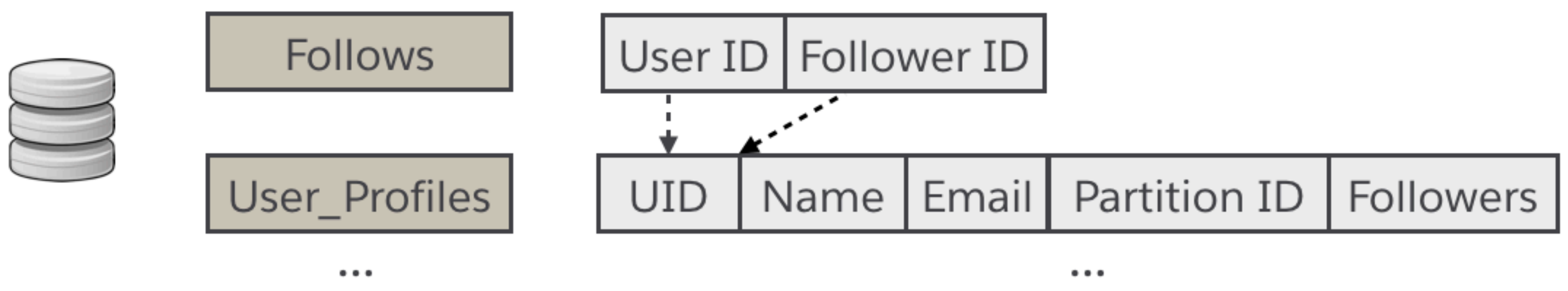
Yu et al. 2018c
Bogin et al. 2019
Guo et al. 2019
Wang et al. 2020
…

Simplified architecture; transfer learning, human-in-the-loop

Lin et al. 2020
Scholak et al. 2020
Zhong et al. 2020
Yao et al. 2020
…

TSP Problem Overview

Domain Twitter



Follows

User ID	Follower ID

User_Profiles

UID	Name	Email	Partition ID	Followers

...

...

List the name and ***number of*** followers for each user

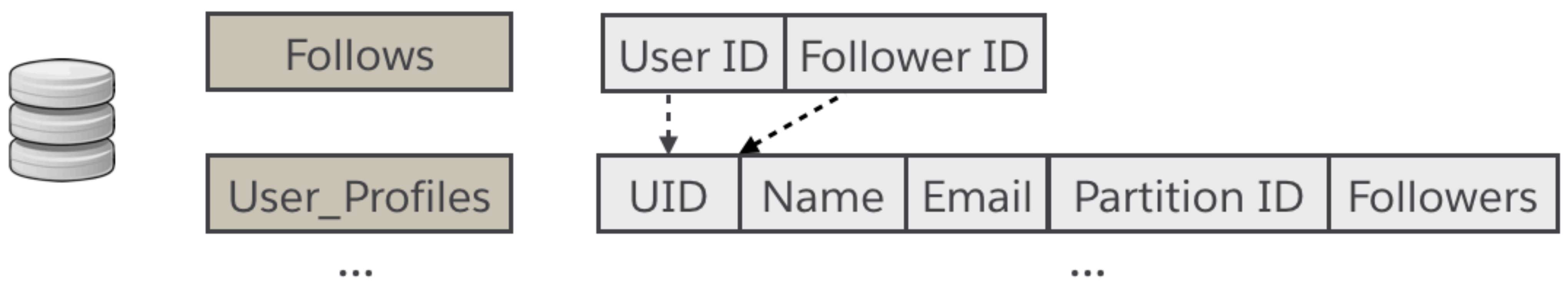
SQL

```
SELECT name, followers FROM User_Profiles
```

Tables are the simplest relational databases

TSP Problem Overview

Domain Twitter



Follows: User ID | Follower ID

User_Profiles: UID | Name | Email | Partition ID | Followers

...

List the name and ***number of*** followers for each user

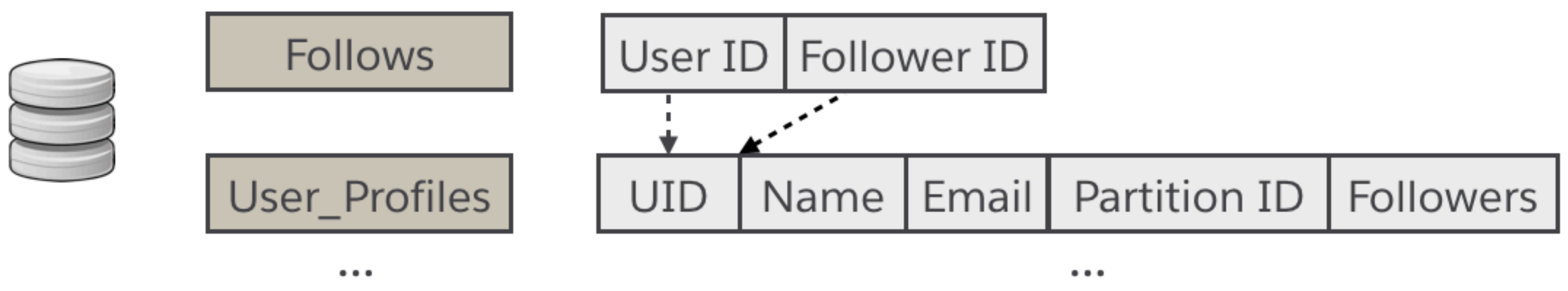
SQL

```
SELECT name, followers FROM User_Profiles
```

Strong/Fully supervised SP

TSP Problem Overview

Domain Twitter



Follows	User ID	Follower ID			
User_Profiles	UID	Name	Email	Partition ID	Followers
...	...				

List the name and ***number of*** followers for each user

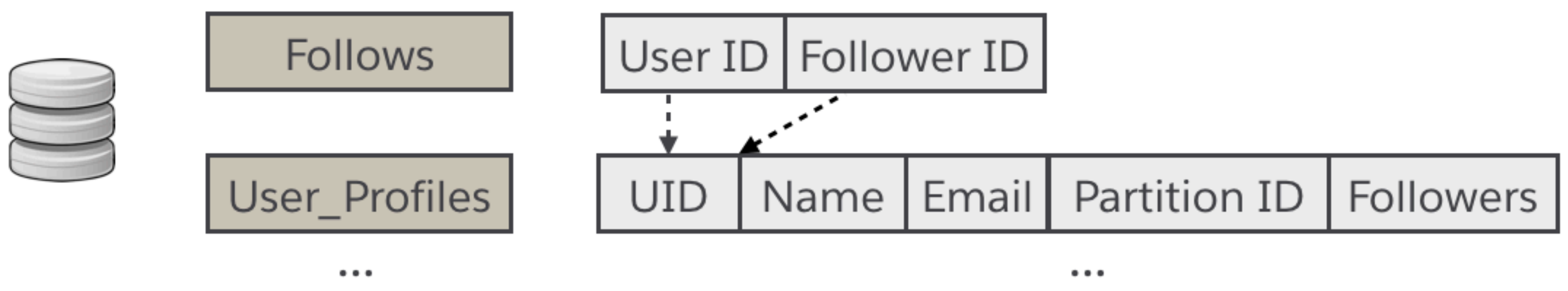
~~SQL~~ ~~**SELECT** name, followers **FROM** User_Profiles~~

Weakly-supervised SP

Learning Executable Semantic Parsers for Natural Language Understanding. Liang et al. 2016.

TSP Problem Overview

Domain Twitter



Follows | User ID | Follower ID

User_Profiles | UID | Name | Email | Partition ID | Followers

… …

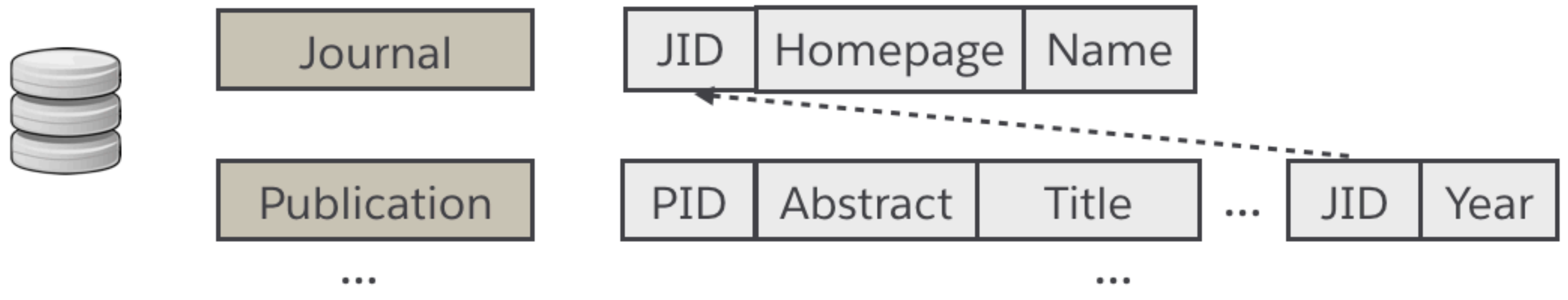
List the name and ***number of*** followers for each user

SQL

```
SELECT name, followers FROM User_Profiles
```

Cross-Database

Domain Academic



Journal | JID | Homepage | Name

Publication | PID | Abstract | Title | … | JID | Year

… …

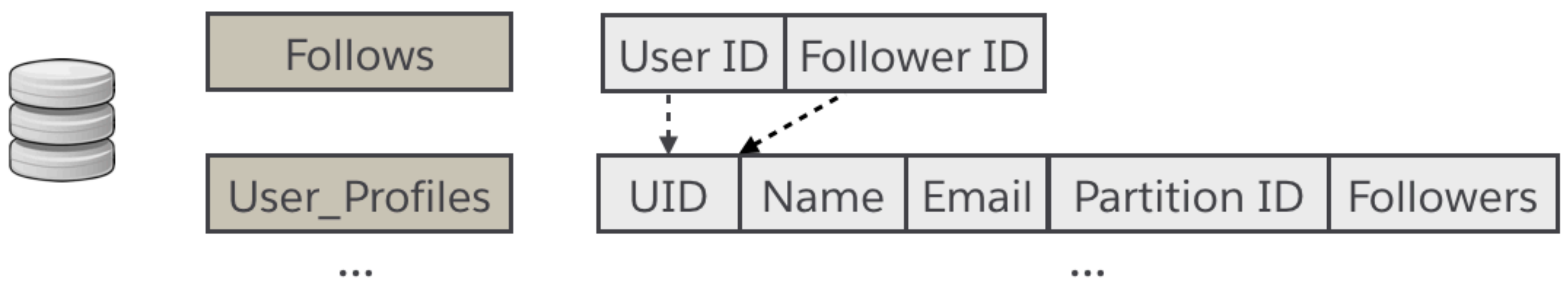
Return me the ***number of*** papers on PVLDB

SQL

```
SELECT COUNT(DISTINCT t2.title)
FROM Publication AS T2 JOIN Journal AS T1
ON T2.JID = T1.JID WHERE T1.name = “PVLDB”
```

TSP Problem Overview

Domain Twitter



Follows

User ID | Follower ID

User_Profiles

UID | Name | Email | Partition ID | Followers

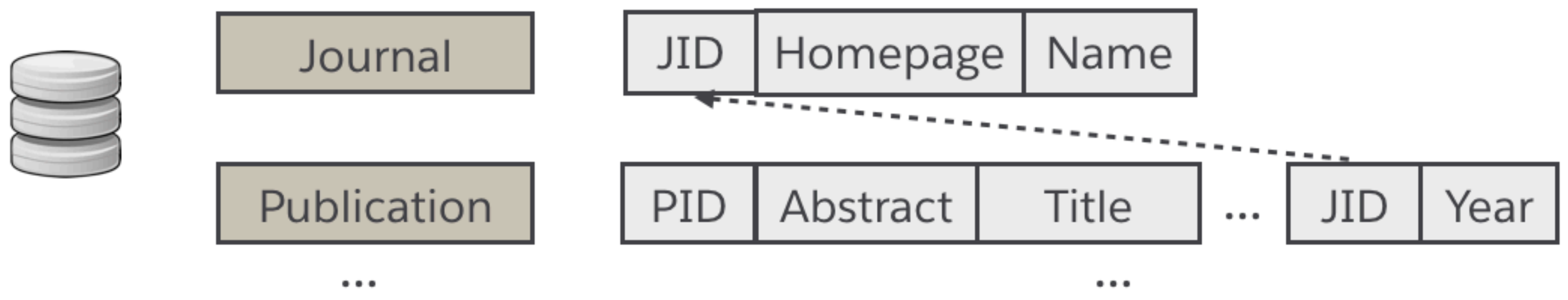
...

List the name and ***number of*** followers for each user

SQL

```
SELECT name, followers FROM User_Profiles
```

Domain Academic



Journal

JID | Homepage | Name

Publication

PID | Abstract | Title ... JID | Year

...

Return me the ***number of*** papers on PVLDB

SQL

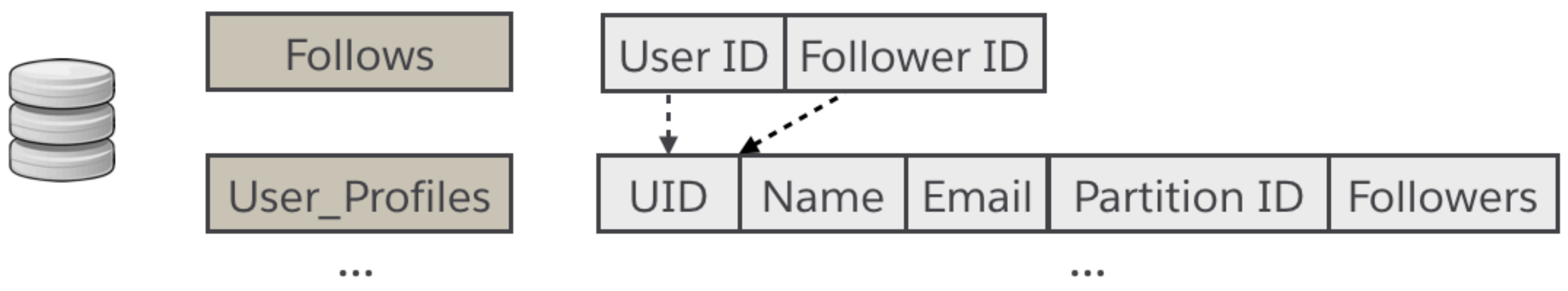
```
SELECT COUNT(DISTINCT t2.title)
FROM Publication AS T2 JOIN Journal AS T1
ON T2.JID = T1.JID WHERE T1.name = "PVLDB"
```

Cross-Database

Similar intent,
different DB schema
results in drastically
different SQL Logical
Forms

TSP Problem Overview

Domain Twitter



Follows

User_Profiles

...

User ID | Follower ID

UID | Name | Email | Partition ID | Followers

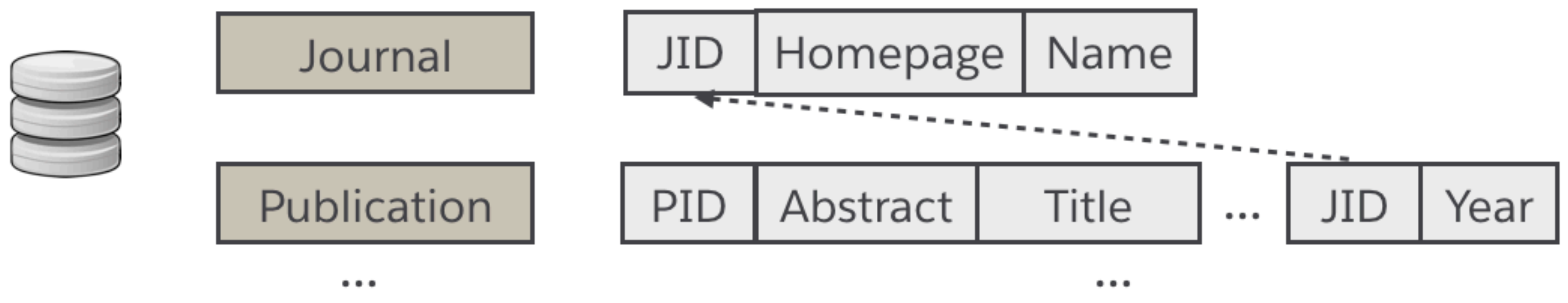
...

List the name and ***number of*** followers for each user

SQL

```
SELECT name, followers FROM User_Profiles
```

Domain Academic



Journal

Publication

...

JID | Homepage | Name

PID | Abstract | Title | ... | JID | Year

...

A long tail of infrequent entity types

Return me the ***number of*** papers on PVLDB

SQL

```
SELECT COUNT(DISTINCT t2.title)
FROM Publication AS T2 JOIN Journal AS T1
ON T2.JID = T1.JID WHERE T1.name = "PVLDB"
```

Leverage value-field mappings in the DB

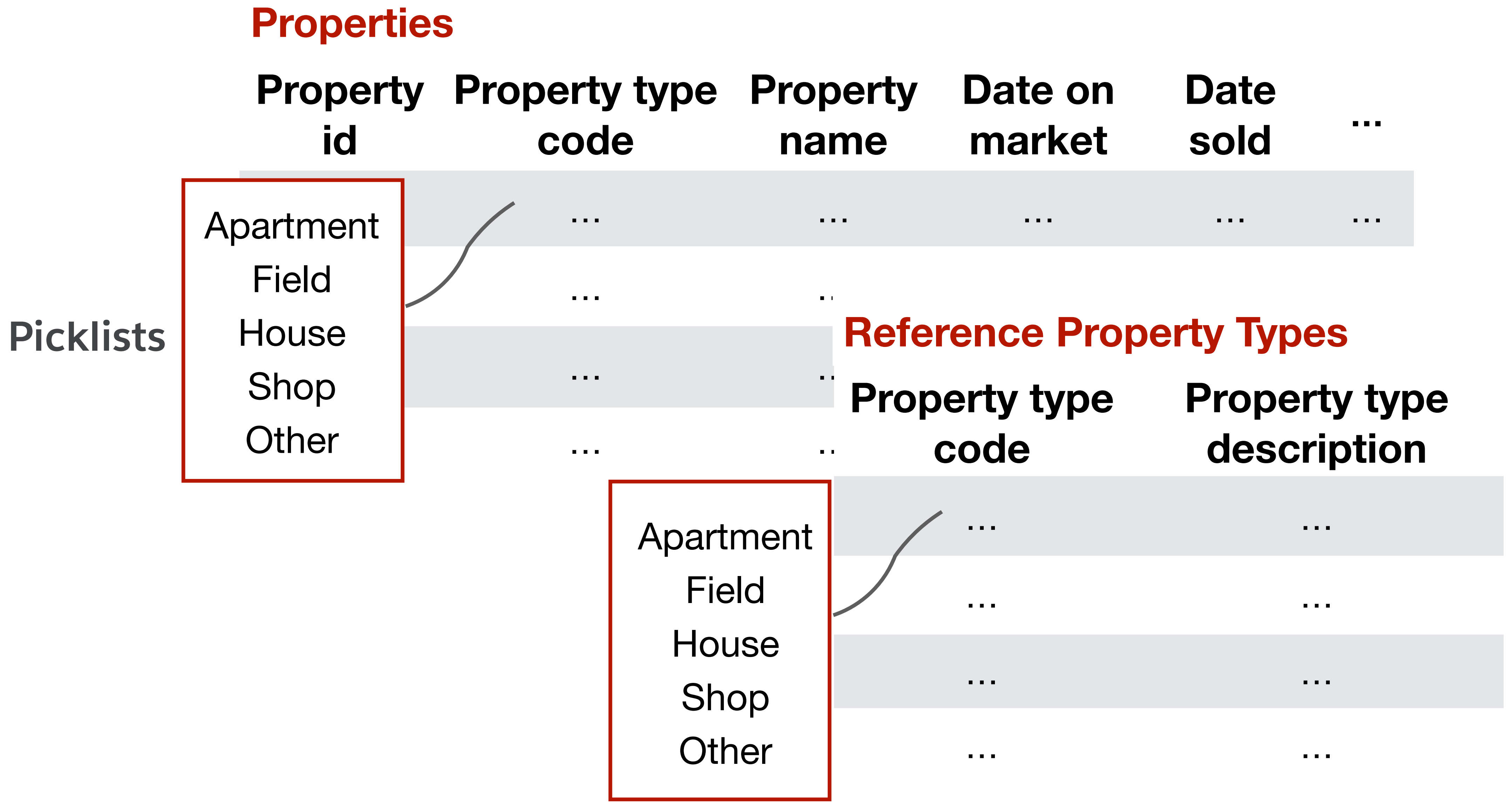
Textual-Tabular Data Encoding

Question

Database Schema

💡 Environment

Show names of properties that are either houses or apartments



Properties

Property id	Property type code	Property name	Date on market	Date sold	...
	...	...	...	...	...
	...	..			
	...	..			
	...	..			

Picklists

Apartment
Field
House
Shop
Other

Reference Property Types

Property type code	Property type description
...	...
...	...
...	...
...	...

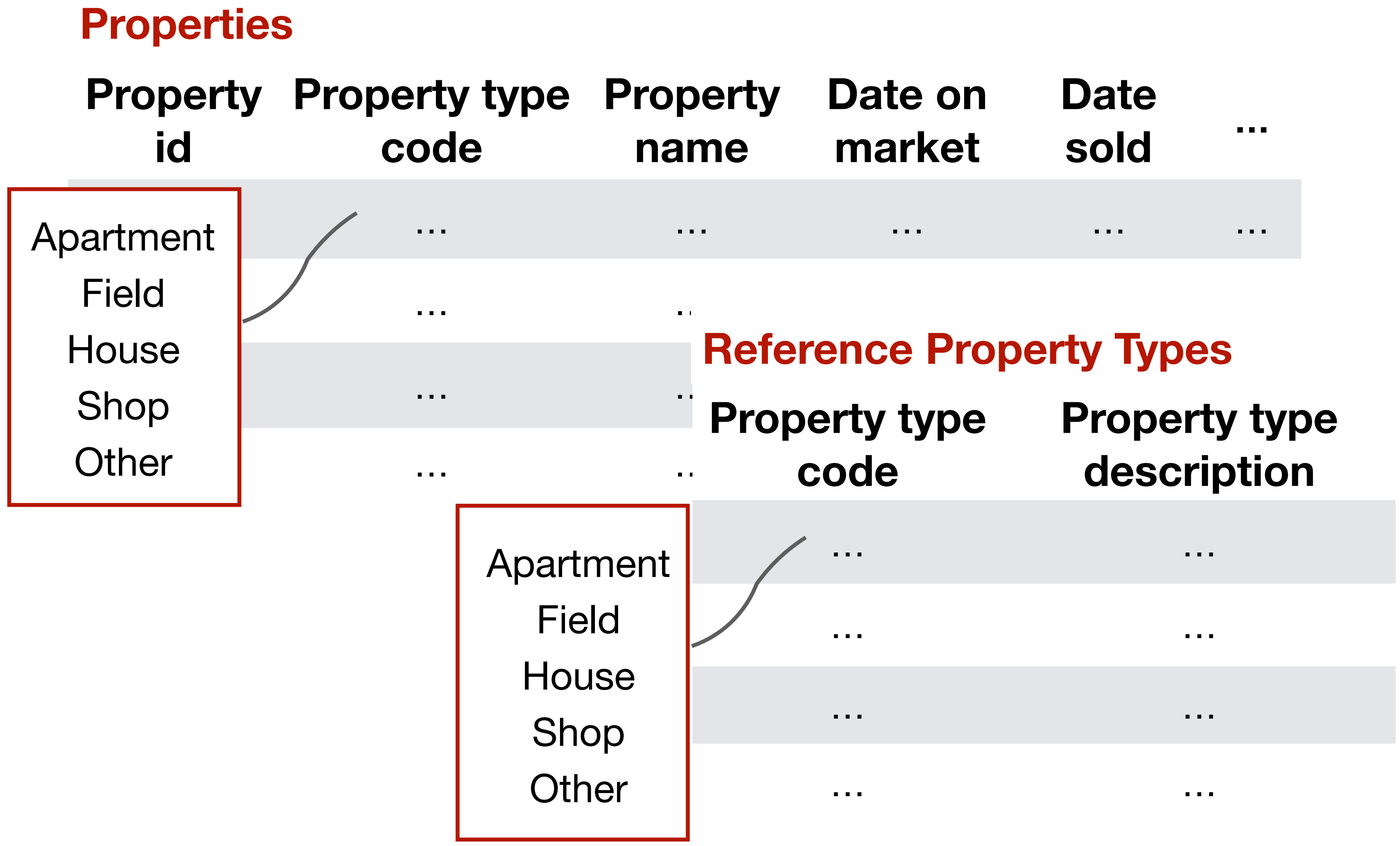
Apartment
Field
House
Shop
Other

Textual-Tabular Data Encoding

Serialize Table Header/DB Schema

CLS	Show names...	SEP	T	Properties	...	C	Property Type Code	C	...	T	Reference Property Types	...	C	Property Type Code	C	...

Show names of properties that are either houses or apartments



Properties

Property id	Property type code	Property name	Date on market	Date sold	...
	...	...	...	...	...
	...	..			
	...	..			
	...	..			

Apartment
Field
House
Shop
Other

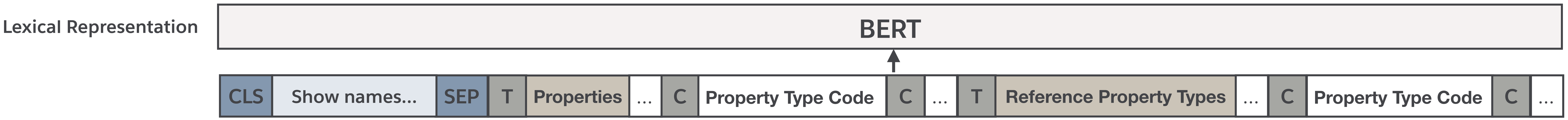
Reference Property Types

Property type code	Property type description
...	...
...	...
...	...
...	...

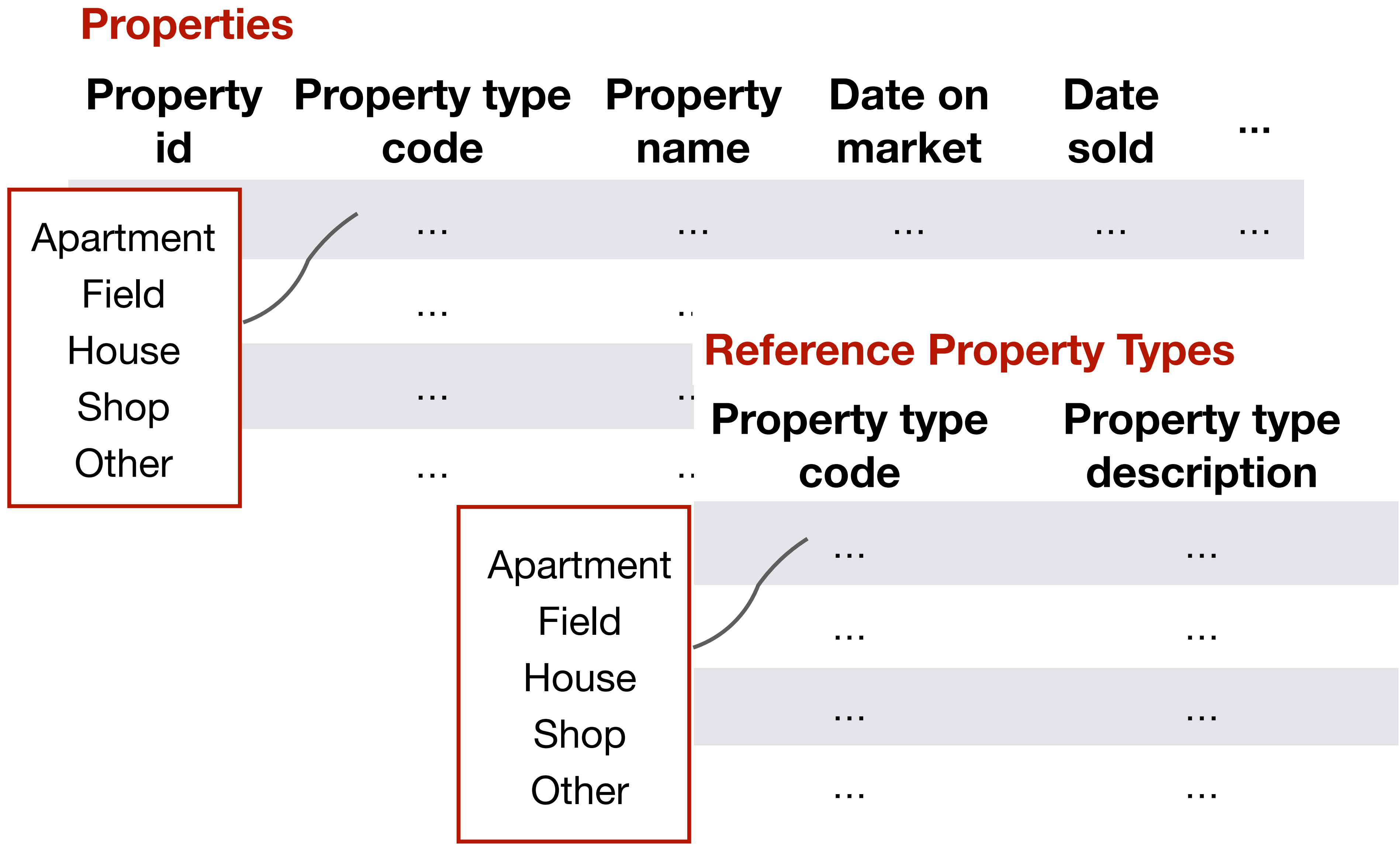
Apartment
Field
House
Shop
Other

Textual-Tabular Data Encoding

Serialize Table Header/DB Schema



Show names of properties that are either houses or apartments


Apartment
Field
House
Shop
Other
Apartment
Field
House
Shop
Other

Properties

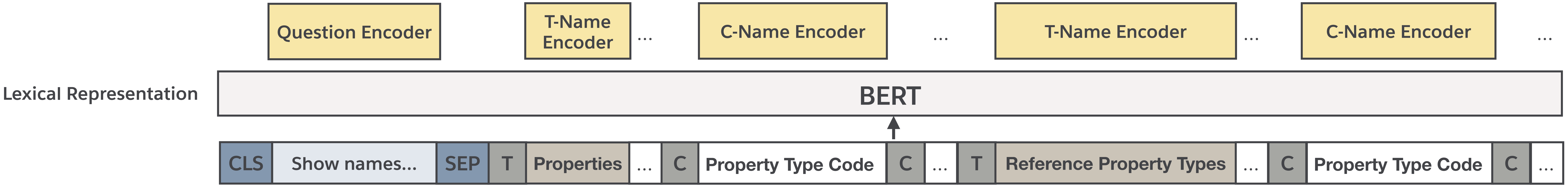
Property id	Property type code	Property name	Date on market	Date sold	...
	...	...	...	...	...
	...	..			
	...	..			
	...	..			

Reference Property Types

Property type code	Property type description
...	...
...	...
...	...
...	...

Textual-Tabular Data Encoding

Separate Question/Table/Field Encoder



Question Encoder

T-Name Encoder ...

C-Name Encoder ...

T-Name Encoder ...

C-Name Encoder ...

Lexical Representation

BERT

CLS	Show names...	SEP	T	Properties	...	C	Property Type Code	C	...	T	Reference Property Types	...	C	Property Type Code	C	...

Show names of properties that are either houses or apartments

Properties

Property id	Property type code	Property name	Date on market	Date sold	...
	...	...	...	...	...
	...	..			
	...	..			
	...	..			

Apartment
Field
House
Shop
Other

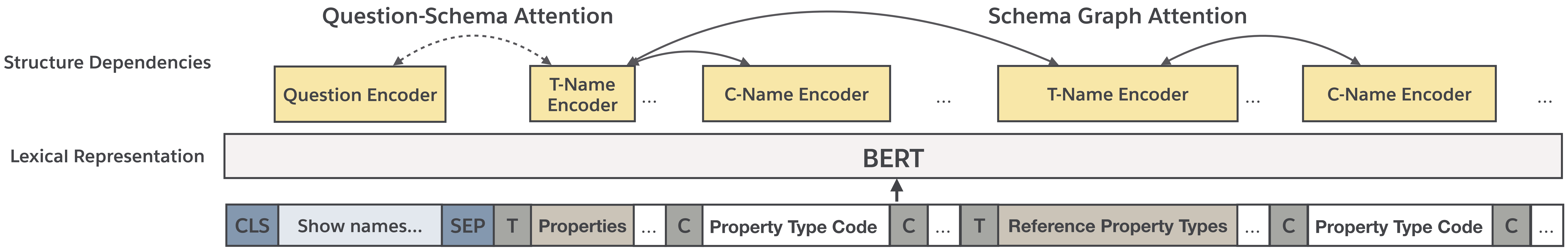
Reference Property Types

Property type code	Property type description
...	...
...	...
...	...
...	...

Apartment
Field
House
Shop
Other

Textual-Tabular Data Encoding

Cross-Component Attention



Show names of properties that are either houses or apartments

Properties

Property id	Property type code	Property name	Date on market	Date sold	...
	...	...	...	...	...
	...	..			
	...	..			
	...	..			

Apartment
Field
House
Shop
Other

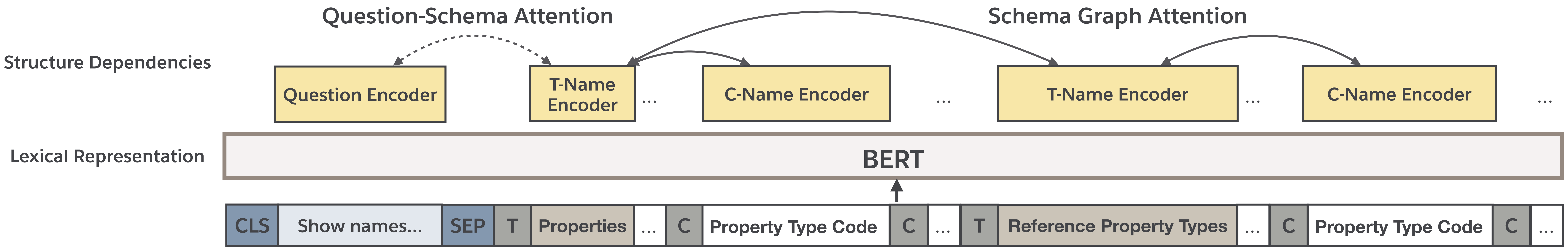
Reference Property Types

Property type code	Property type description
...	...
...	...
...	...
...	...

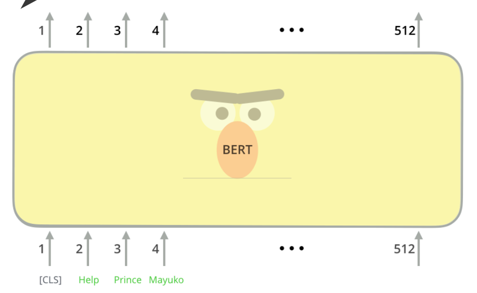
Apartment
Field
House
Shop
Other

Textual-Tabular Data Encoding

Cross-Component Attention



Architecture Redundancy



Show names of properties that are either houses or apartments

Properties

Property id	Property type code	Property name	Date on market	Date sold	…
	…	…	…	…	…
	…	..			
	…	..			
	…	..			

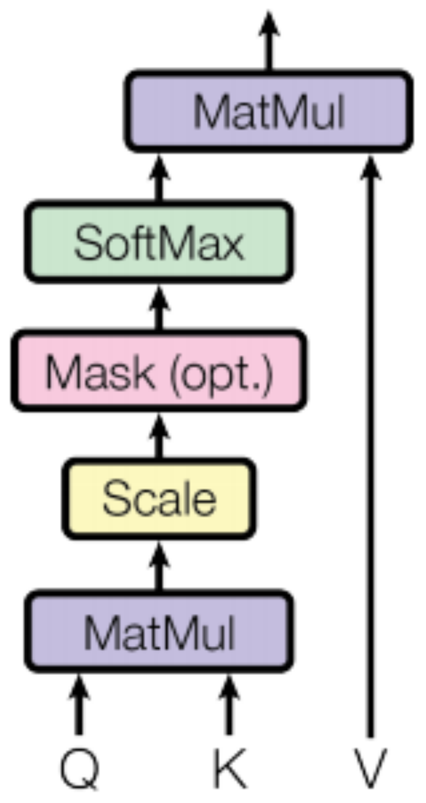
Apartment
Field
House
Shop
Other

Reference Property Types

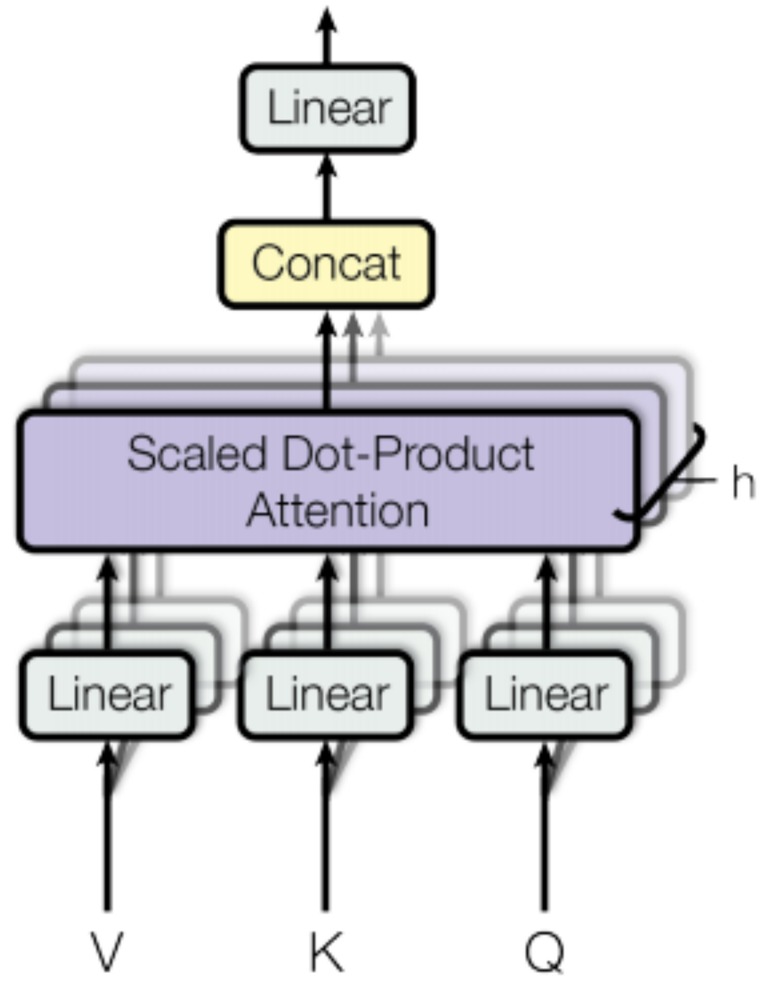
Property type code	Property type description
…	…
…	…
…	…
…	…

Apartment
Field
House
Shop
Other

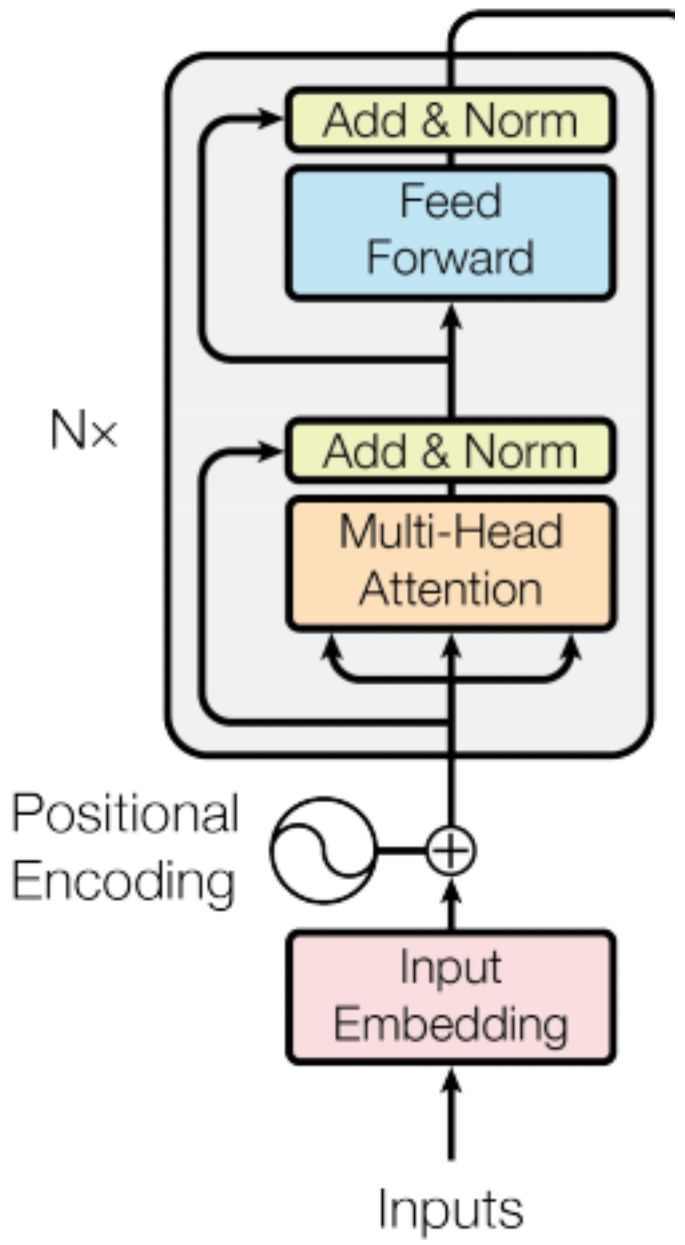
Recap: Attention



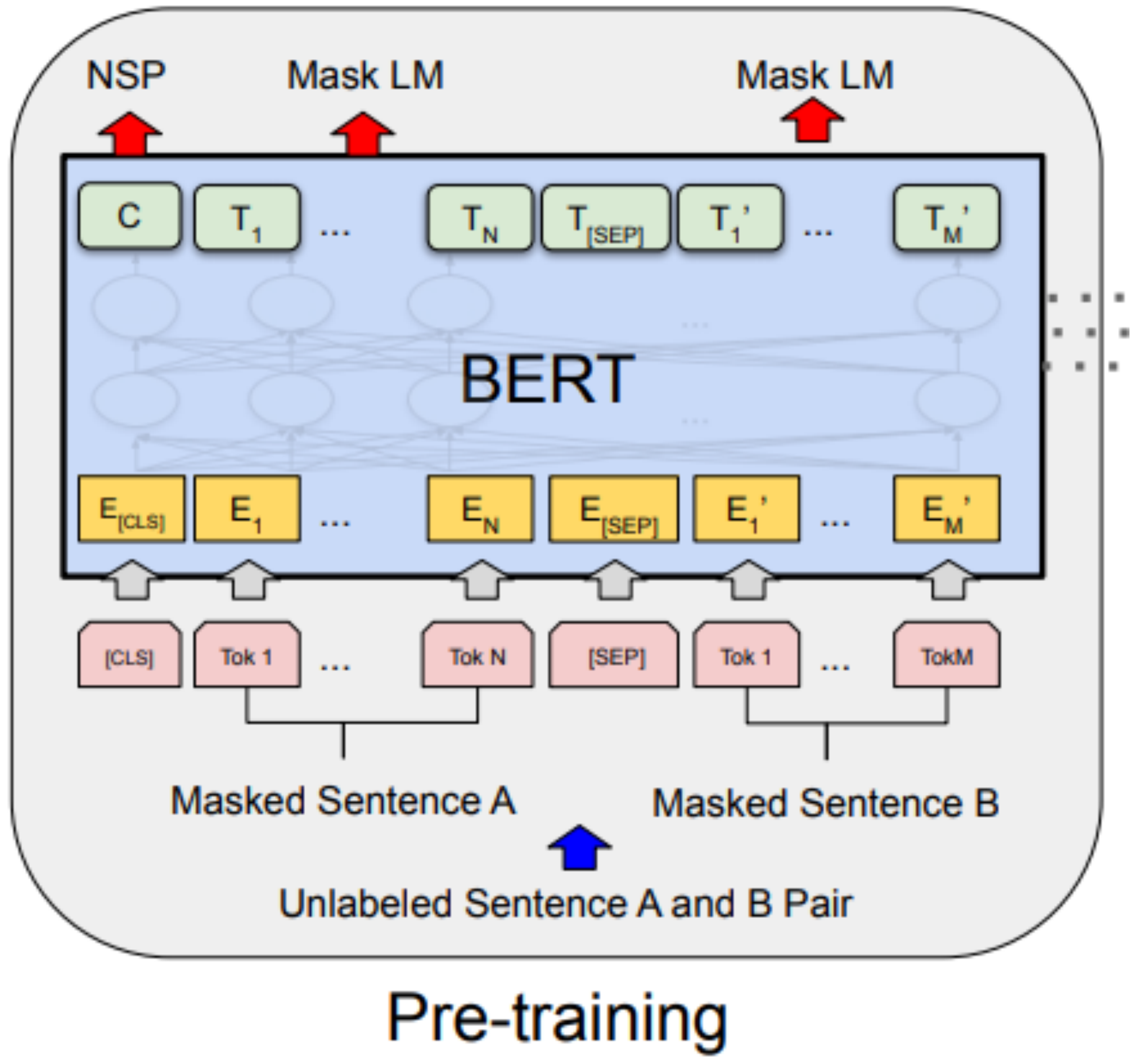
(A) Scaled Dot-Product Attention



(B) Multi-Head Attention



(C) Self-Attention (Encoder Representations from Transformers)

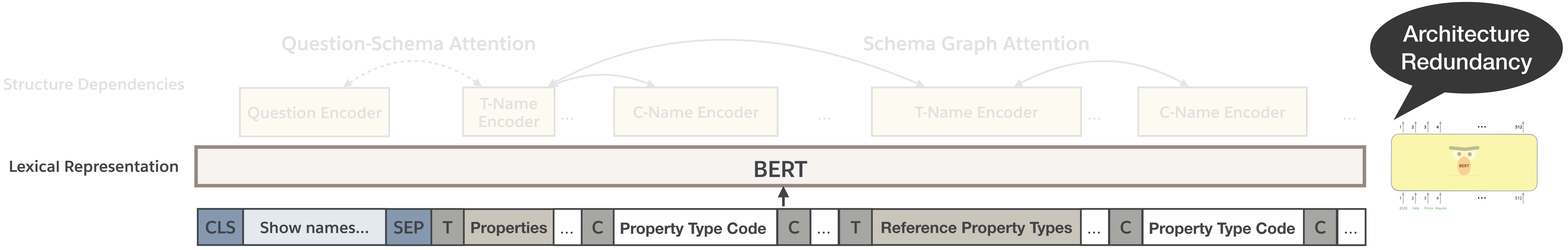


(D) Pre-trained Bidirectional Encoder Representations from Transformers (BERT)

Attention Is All You Need. Vaswani et. al. 2017.

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. Devlin et. al. 2018.

Textual-Tabular Data Encoding



Show names of properties that are either houses or apartments

Idea: Encode question, DB schema and the cross-modal contextualization using pre-trained deep BERT.

Properties

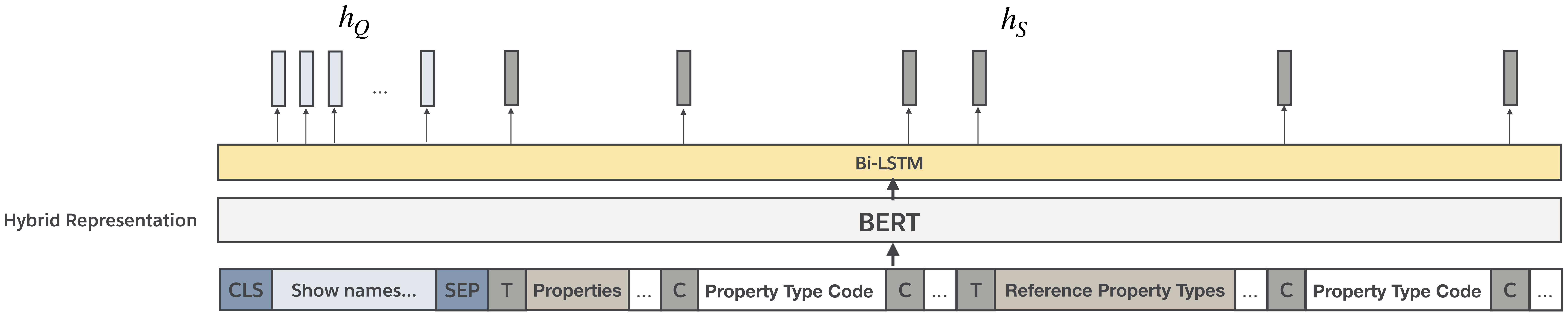
Property id	Property type code	Property name	Date on market	Date sold	…
Apartment Field House Shop Other	…	…	…	…	…
	…				
	…				
	…				

Reference Property Types

Property type code	Property type description
Apartment Field House Shop Other …	…
…	…
…	…
…	…

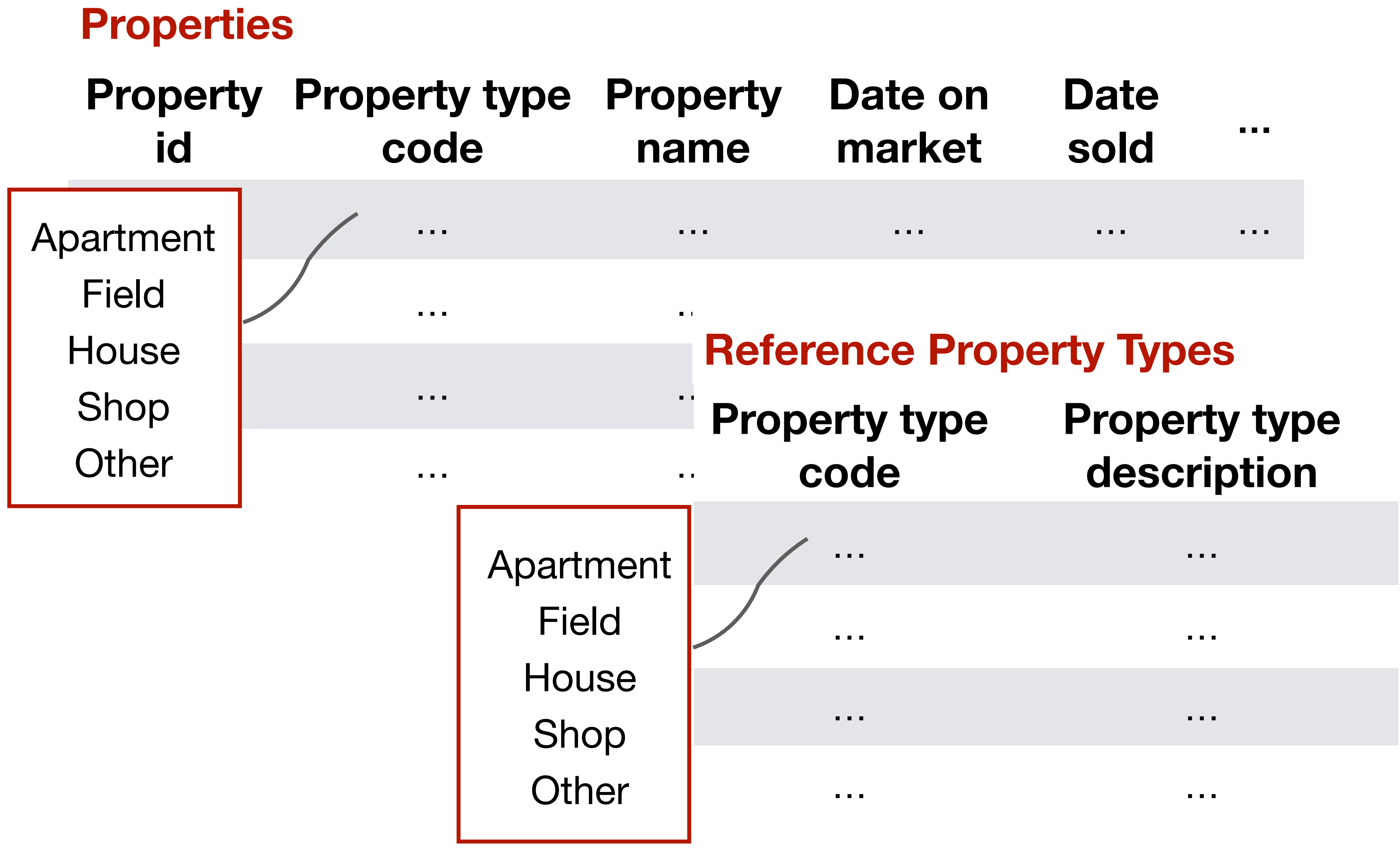
Bridging Textual and Tabular Data for Cross-Domain Text-to-SQL Semantic Parsing. Lin et. al. 2020.

Textual-Tabular Data Encoding



Show names of properties that are either houses or apartments

Idea: Encode question, DB schema and the cross-modal contextualization using pre-trained deep BERT.


Properties

Property id	Property type code	Property name	Date on market	Date sold	...
	...	...	...	...	...
	...	..			
	...	..			
	...	..			

Apartment
Field
House
Shop
Other

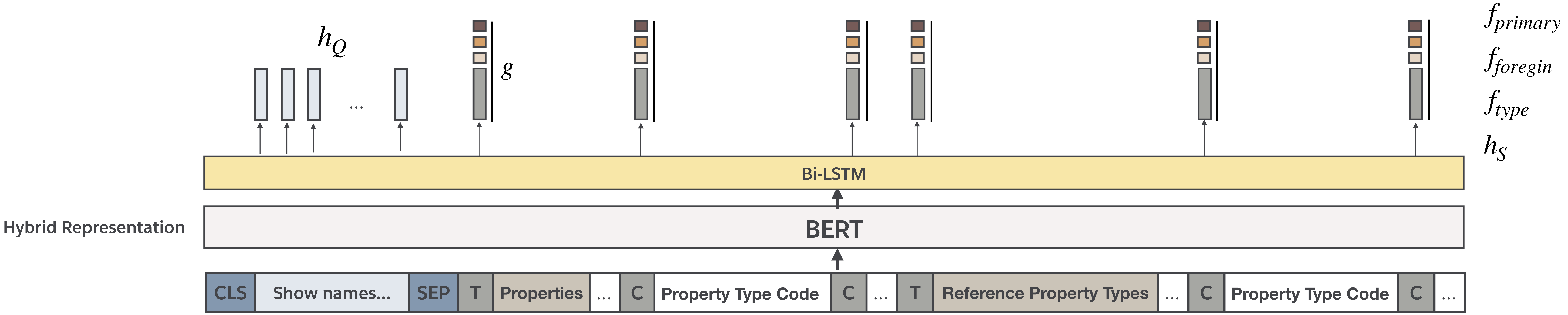
Reference Property Types

Property type code	Property type description
...	...
...	...
...	...
...	...

Apartment
Field
House
Shop
Other

Bridging Textual and Tabular Data for Cross-Domain Text-to-SQL Semantic Parsing. Lin et. al. 2020.

Textual-Tabular Data Encoding



Show names of properties that are either houses or apartments

Properties

Property id	Property type code	Property name	Date on market	Date sold	...
	...	...	...	...	...
	...	..			
	...	..			
	...	..			

Apartment
Field
House
Shop
Other

Reference Property Types

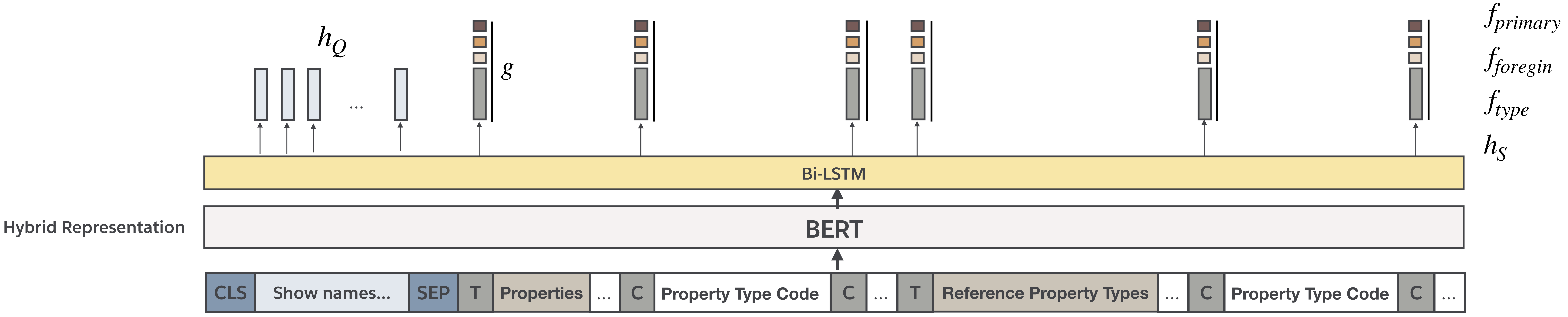
Property type code	Property type description
...	...
...	...
...	...
...	...

Apartment
Field
House
Shop
Other

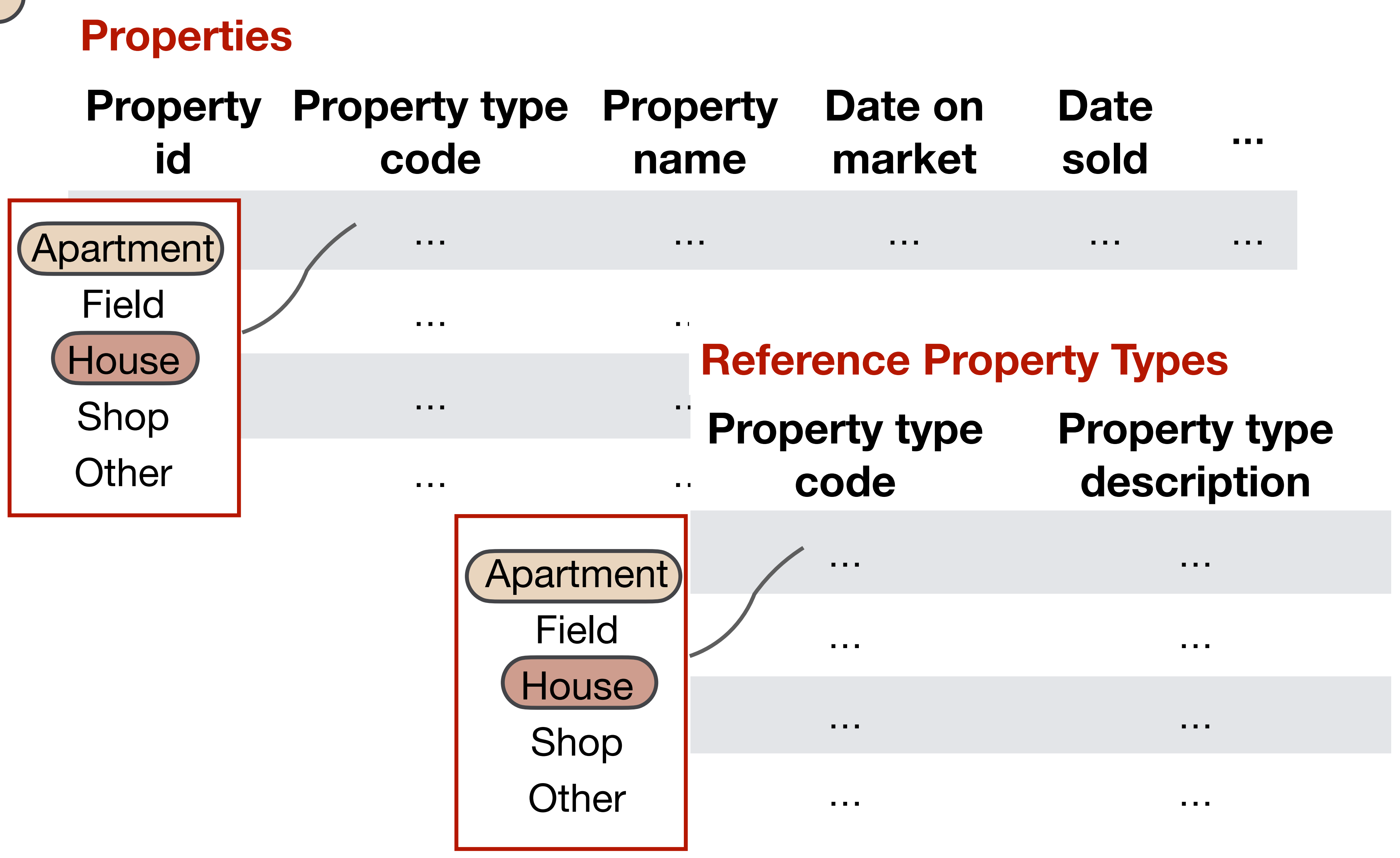
Idea: Encode question, DB schema and the cross-modal contextualization using pre-trained deep BERT.

Bridging Textual and Tabular Data for Cross-Domain Text-to-SQL Semantic Parsing. Lin et. al. 2020.

Bridging



Show names of properties that are either houses or apartments



Properties

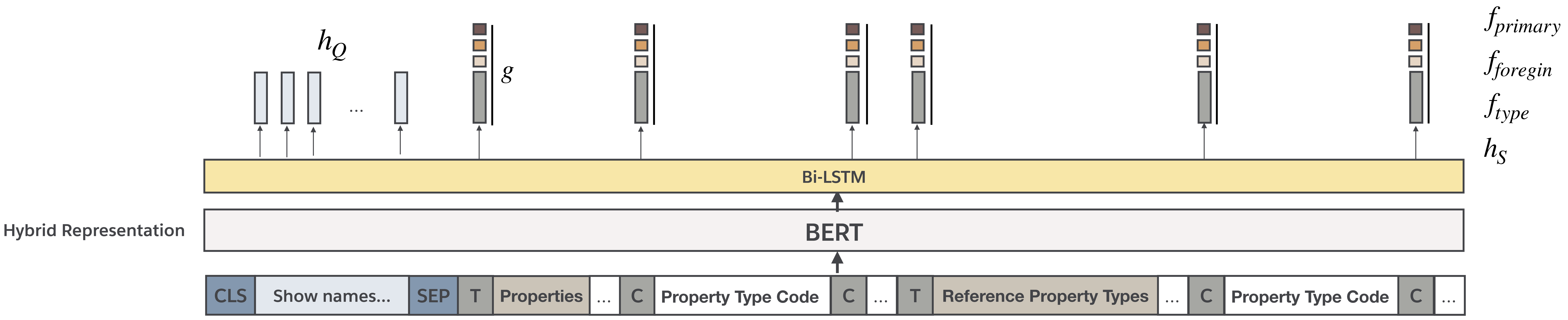
Property id	Property type code	Property name	Date on market	Date sold	...
	...	...	...	...	...
	...	..			
	...	..			
	...	..			

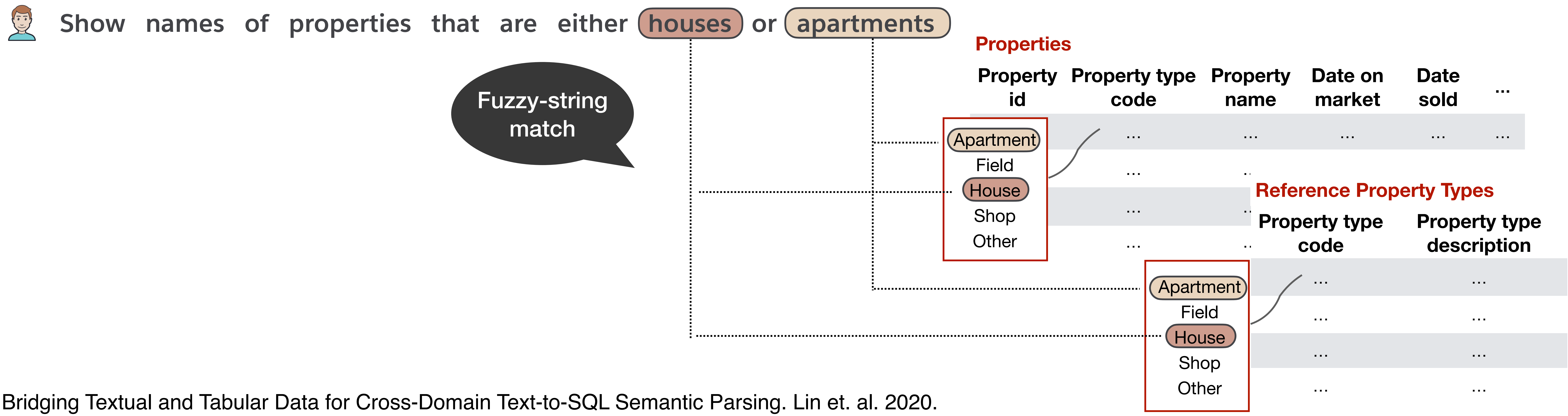
Reference Property Types

Property type code	Property type description
...	...
...	...
...	...
...	...

Bridging Textual and Tabular Data for Cross-Domain Text-to-SQL Semantic Parsing. Lin et. al. 2020.

Bridging




Show names of properties that are either houses or apartments
Fuzzy-string match

Properties

Property id	Property type code	Property name	Date on market	Date sold	...
	...	...	...	...	...
	...	..			
	...	..			
	...	..			

Apartment
Field
House
Shop
Other

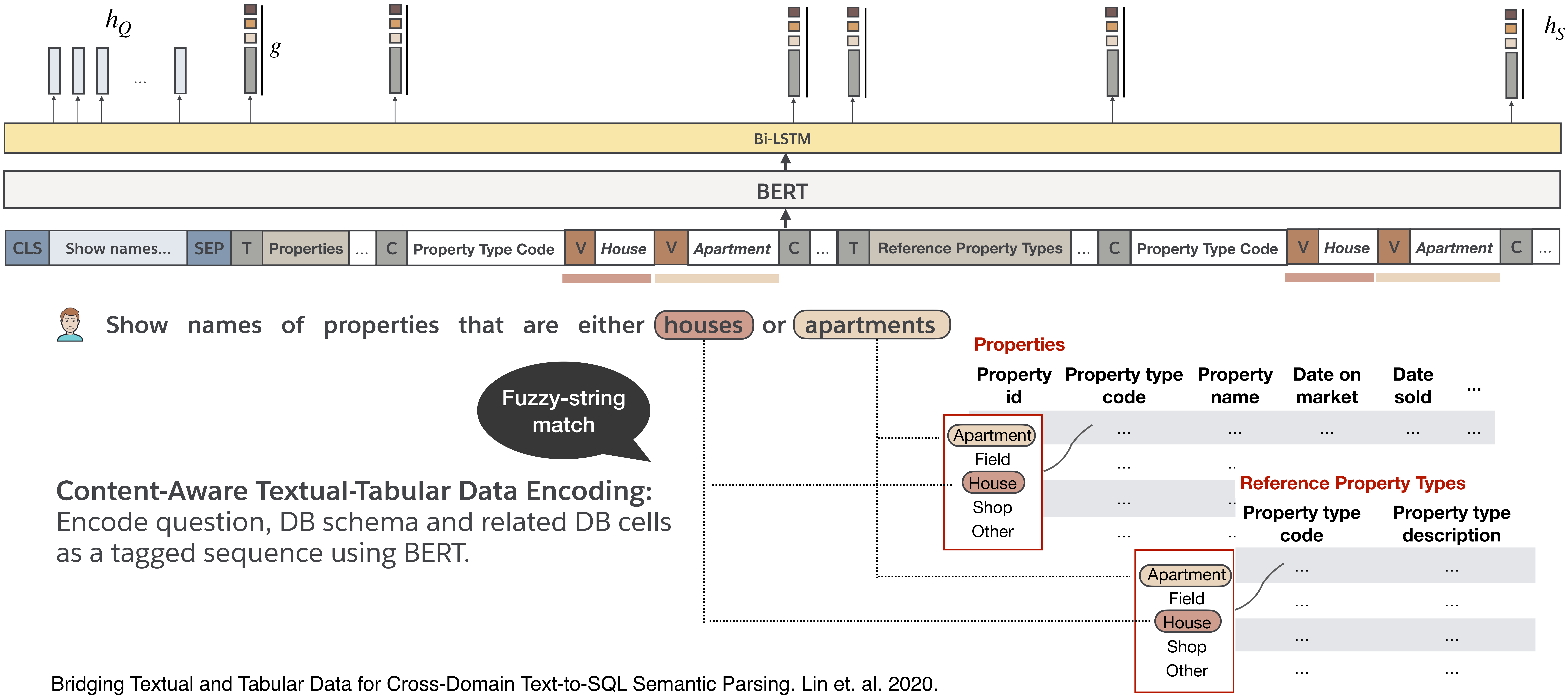
Reference Property Types

Property type code	Property type description
...	...
...	...
...	...
...	...

Apartment
Field
House
Shop
Other

Bridging Textual and Tabular Data for Cross-Domain Text-to-SQL Semantic Parsing. Lin et. al. 2020.

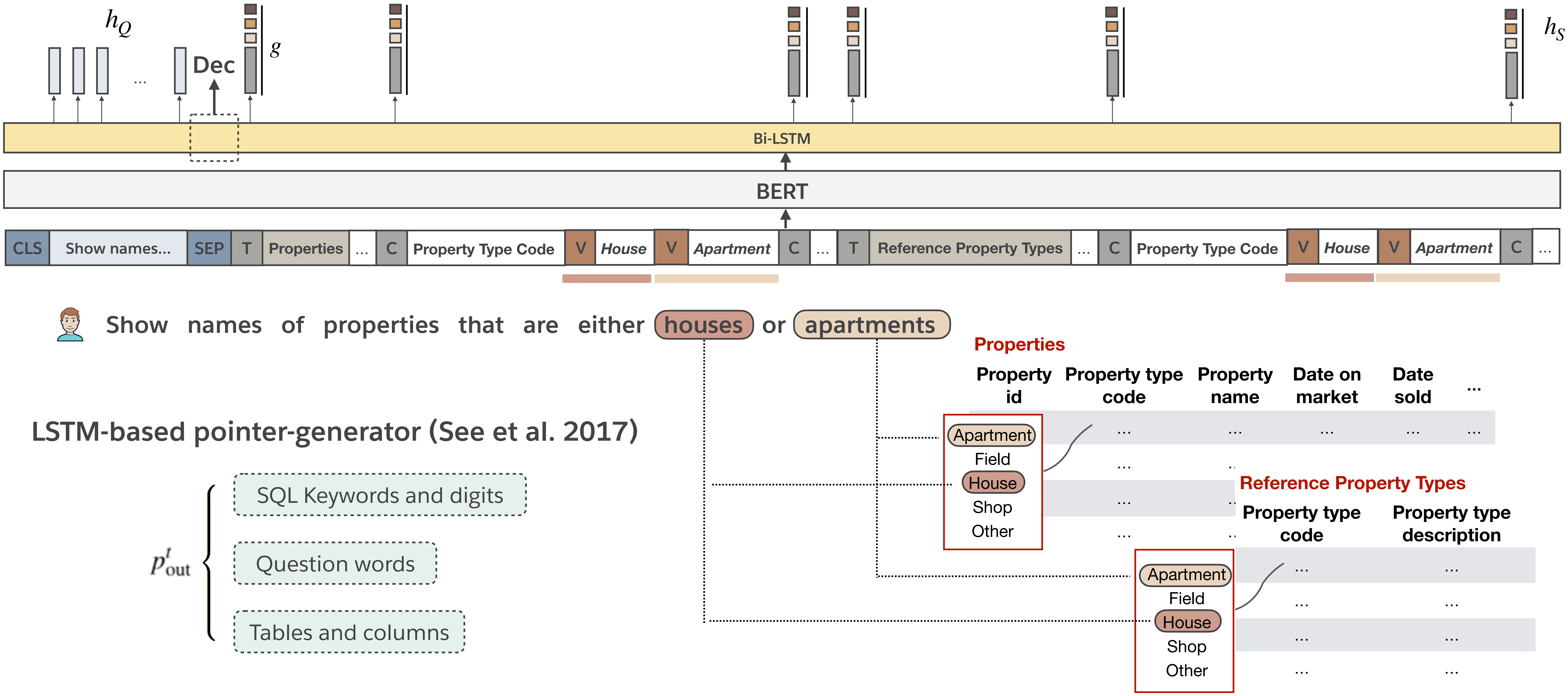
Bridging



Content-Aware Textual-Tabular Data Encoding:
Encode question, DB schema and related DB cells as a tagged sequence using BERT.

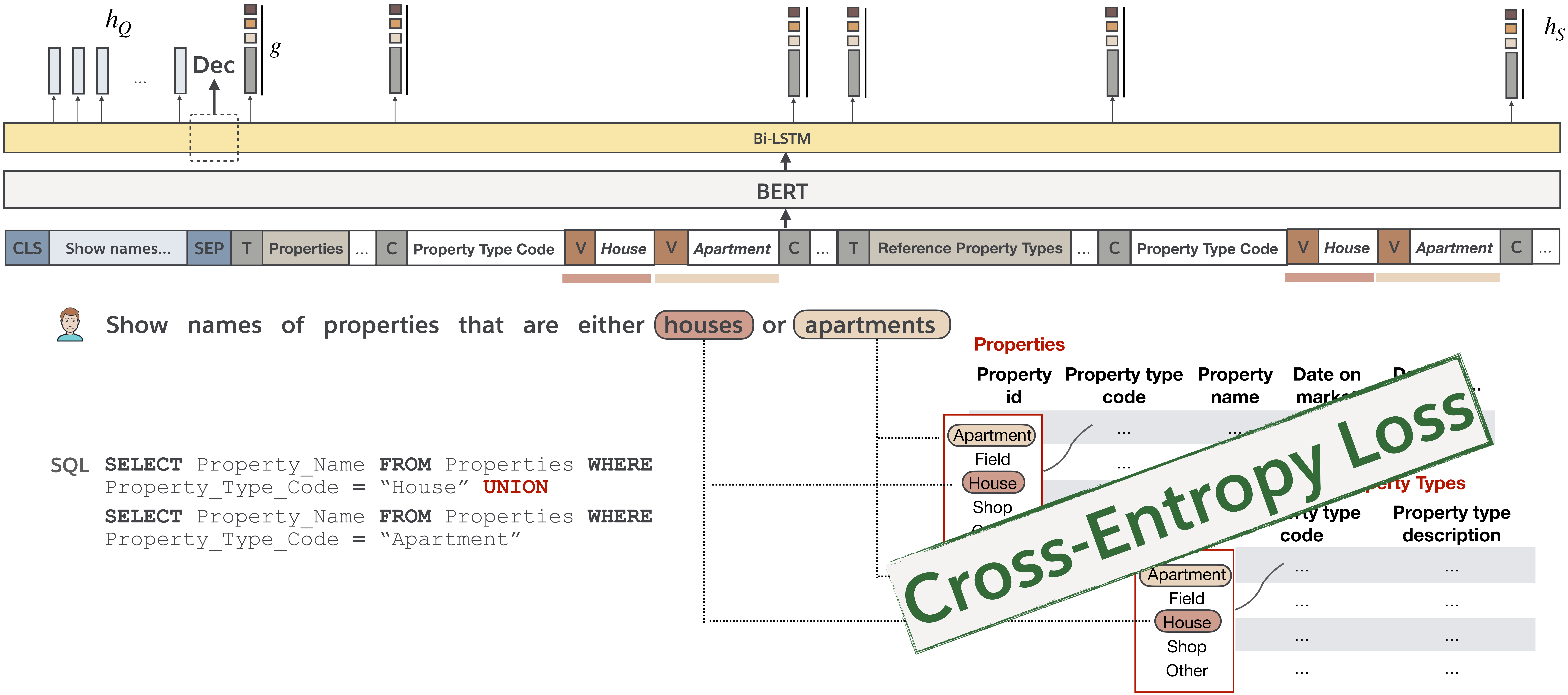
Bridging Textual and Tabular Data for Cross-Domain Text-to-SQL Semantic Parsing. Lin et. al. 2020.

Decoder



LSTM-based pointer-generator (See et al. 2017)

Decoder



Show names of properties that are either houses or apartments

SQL
```
SELECT Property_Name FROM Properties WHERE
Property_Type_Code = "House" UNION
SELECT Property_Name FROM Properties WHERE
Property_Type_Code = "Apartment"
```

Cross-Entropy Loss

Schema-Consistency Guided Decoding

Effective heuristics for pruning the search space of a sequential pointer-generator decoder

- SQL syntax

Schema-Consistency Guided Decoding

Effective heuristics for pruning the search space of a sequential pointer-generator decoder

- SQL syntax
- The FROM clauses set the scope of a SQL query and the table fields appeared in the rest of the clauses can only belong to the tables in FROM

```
SELECT T2.name FROM Instructor AS T1 JOIN Department AS T2 ON T1.Department_ID = T2.ID
GROUP BY T1.Department_ID HAVING AVG(T1.Rating) > (SELECT AVG(Rating) FROM Instructor)
```

Schema-Consistency Guided Decoding

Effective heuristics for pruning the search space of a sequential pointer-generator decoder

- SQL syntax
- The FROM clauses set the scope of a SQL query and the table fields appeared in the rest of the clauses can only belong to the tables in FROM

```
SELECT T2.name FROM Instructor AS T1 JOIN Department AS T2 ON T1.Department_ID = T2.ID
GROUP BY T1.Department_ID HAVING AVG(T1.Rating) > (SELECT AVG(Rating) FROM Instructor)
```

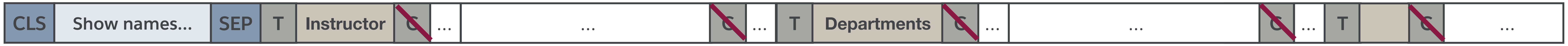
Rewrite a SQL query with FROM clauses in the front execution order

```
FROM Instructor AS T1 JOIN Department AS T2 ON T1.Department_ID = T2.ID SELECT T2.name
GROUP BY T1.Department_ID HAVING AVG(T1.Rating) > (FROM Instructor SELECT AVG(Rating))
```

Lemma: Let Y_{exec} be a SQL query with clauses arranged in execution order, then any table field in Y_{exec} will appear after its corresponding table token.

Schema-Consistency Guided Decoding

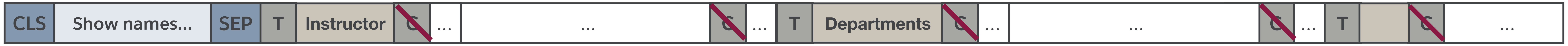
- Generate SQL queries in execution order and unmask DB fields dynamically



CLS	Show names...	SEP	T	Instructor	~~C~~	...	...	~~C~~	...	T	Departments	~~C~~	...	...	~~C~~	...	T		~~C~~	...	

Schema-Consistency Guided Decoding

- Generate SQL queries in execution order and unmask DB fields dynamically

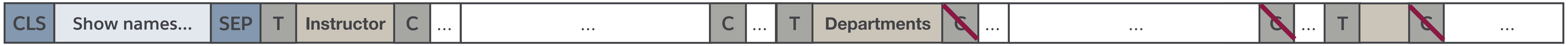


CLS | Show names... | SEP | T | Instructor | ~~C~~ | ... | ... | ~~C~~ | ... | T | Departments | ~~C~~ | ... | ... | ~~C~~ | ... | T | | ~~C~~ | ...

```
FROM
```

Schema-Consistency Guided Decoding

- Generate SQL queries in execution order and unmask DB fields dynamically

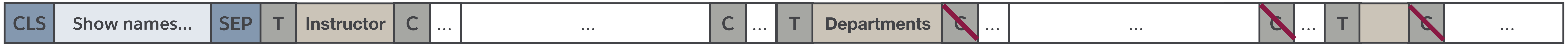


CLS | Show names… | SEP | T | Instructor | C | … | … | C | … | T | Departments | ~~C~~ | … | … | ~~C~~ | … | T | | ~~C~~ | …

```
FROM Instructor
```

Schema-Consistency Guided Decoding

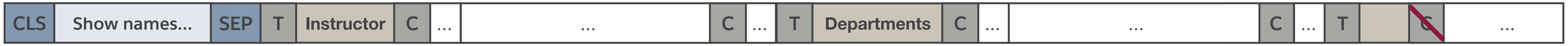
- Generate SQL queries in execution order and unmask DB fields dynamically



```
FROM Instructor JOIN
```

Schema-Consistency Guided Decoding

- Generate SQL queries in execution order and unmask DB fields dynamically

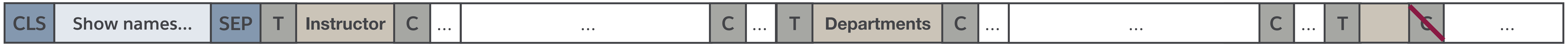


CLS	Show names...	SEP	T	Instructor	C	...	...	C	...	T	Departments	C	...	...	C	...	T		~~C~~	...

```
FROM Instructor JOIN Department
```

Schema-Consistency Guided Decoding

- Generate SQL queries in execution order and unmask DB fields dynamically

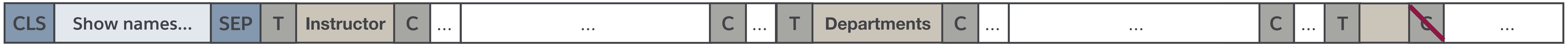


CLS	Show names...	SEP	T	Instructor	C	...	...	C	...	T	Departments	C	...	...	C	...	T		~~C~~	...

```
FROM Instructor JOIN Department ON Instructor.Department_ID = Department.ID SELECT
Department.name GROUP BY Instructor.Department_ID HAVING AVG(Instructor.Rating) >
(FROM Instructor SELECT AVG(Instructor.Rating))
```

Schema-Consistency Guided Decoding

- Generate SQL queries in execution order and unmask DB fields dynamically



CLS	Show names...	SEP	T	Instructor	C	...	...	C	...	T	Departments	C	...	...	C	...	T		~~C~~	...

```
FROM Instructor JOIN Department ON Instructor.Department_ID = Department.ID SELECT
Department.name GROUP BY Instructor.Department_ID HAVING AVG(Instructor.Rating) >
(FROM Instructor SELECT AVG(Instructor.Rating))
```

✓ Vectorizable

✓ Applied during inference

✓ Schema consistency not guaranteed, used in combination with post-decoding checks

✓ Not limited to sequence decoders

Dataset

Spider (Yu et al. 2018)

Expert-annotated, **cross-domain**, **complex** text-to-SQL dataset

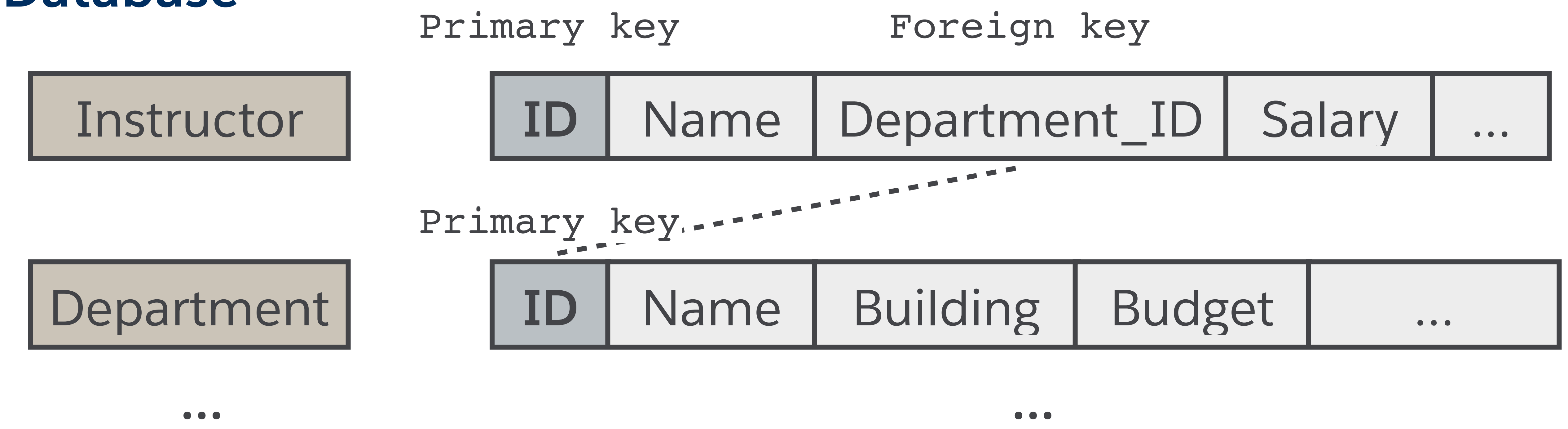
Assumption:

- For each

Hidden

	Train	Dev	Test
# DBs	146	20	40
# Examples	8,659	1,034	2,147

Database



Primary key

Foreign key

Instructor

ID	Name	Department_ID	Salary	...

Primary key

Department

ID	Name	Building	Budget	...

...

...

Question What are the name and budget of the departments with average instructor salary above the overall average?

SQL

```
SELECT T2.name, T2.budget
FROM Instructor AS T1 JOIN Department AS T2 ON
T1.Department_ID = T2.ID
GROUP BY T1.Department_ID
HAVING AVG(T1.salary) >
     (SELECT AVG(Salary) FROM Instructor)
```

Experiments

Inference steps

- Compute fuzzy string match between the input question and the picklists of each DB field to identify value mentions
- For each DB field, keep top-K matches and use them to augment the DB schema representation
- Run semantic parser

Evaluation

- Exact set match
  - Logical form match ignoring values and SQL component order invariance
- Execution accuracy
  - Check if the execution results of the predicted SQL query matches the executions results of the ground-truth SQL query

Experiments

Inference steps

- Compute fuzzy string match between the input question and the picklists of each DB field to identify value mentions
- For each DB field, keep top-K matches and use them to augment the DB schema representation
- Run semantic parser

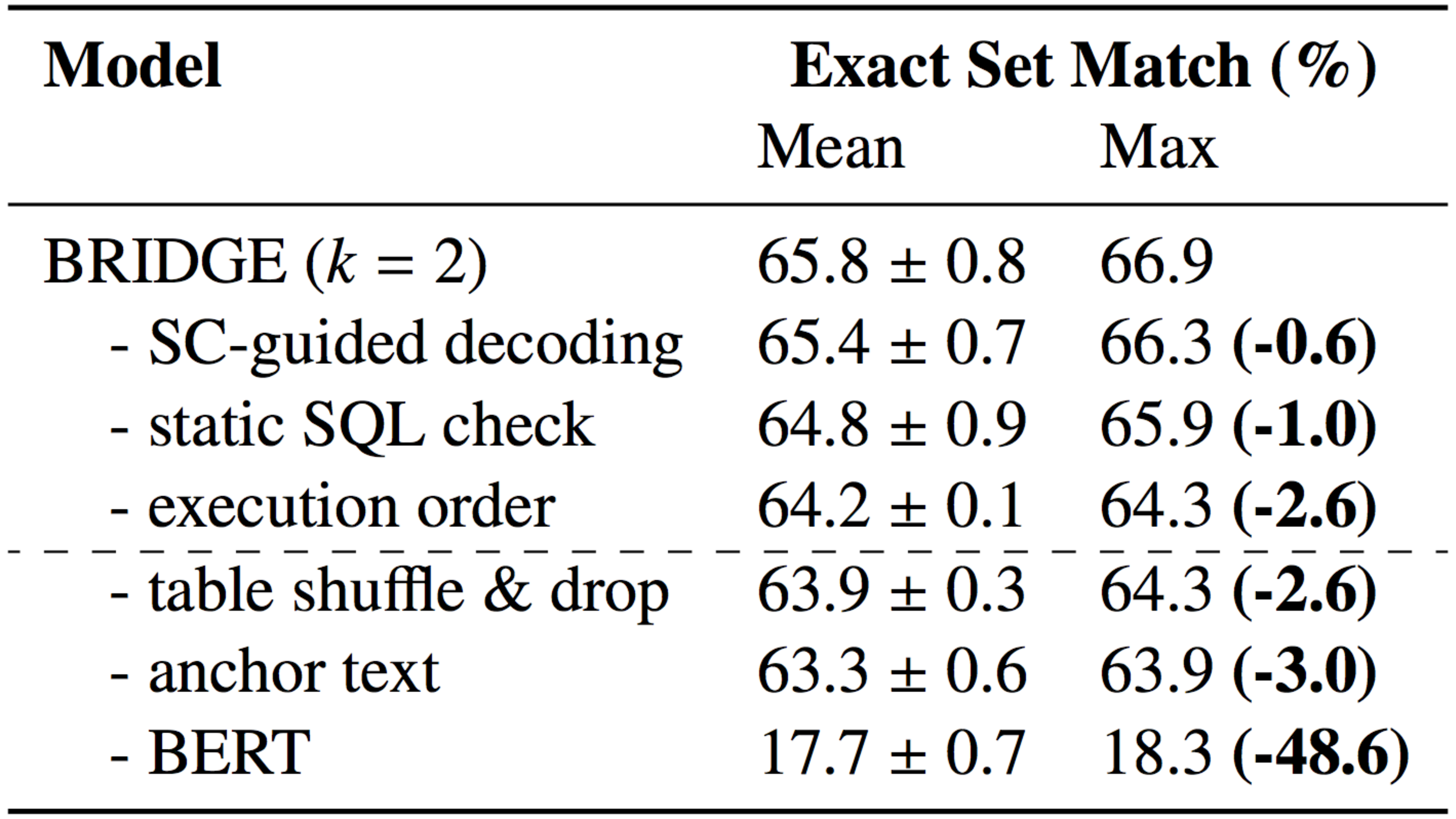
Evaluation

- Exact set match
  - Logical form match ignoring values and SQL component order invariance
- Execution accuracy
  - Check if the execution results of the predicted SQL query matches the executions results of the ground-truth SQL query

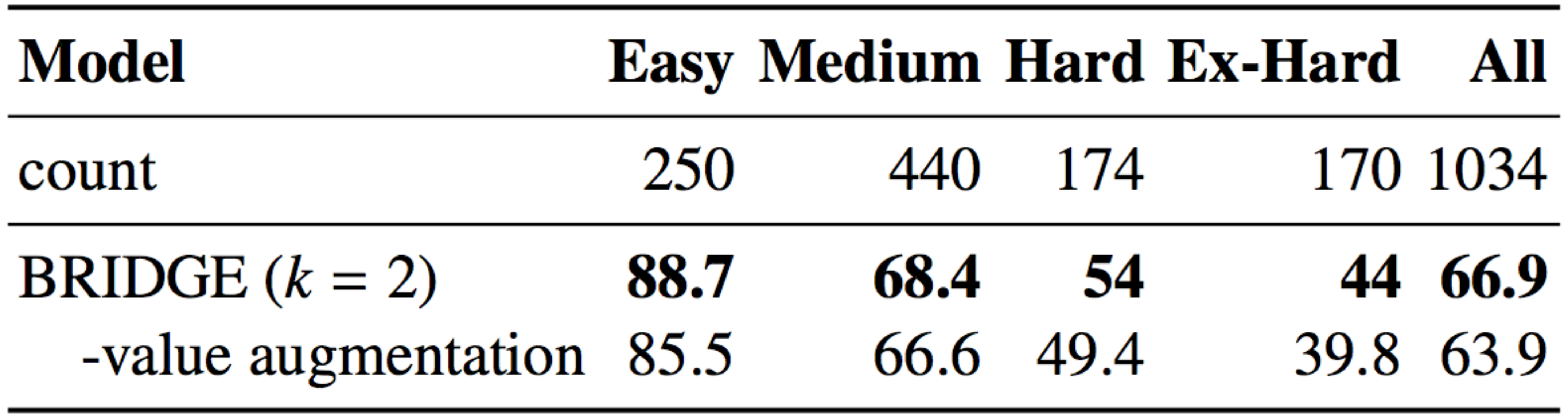
💡 **Better evaluation for text-to-SQL is still an open research problem**

Semantic Evaluation for Text-to-SQL with Distilled Test Suites. Zhong et al. 2020.

Ablation Study

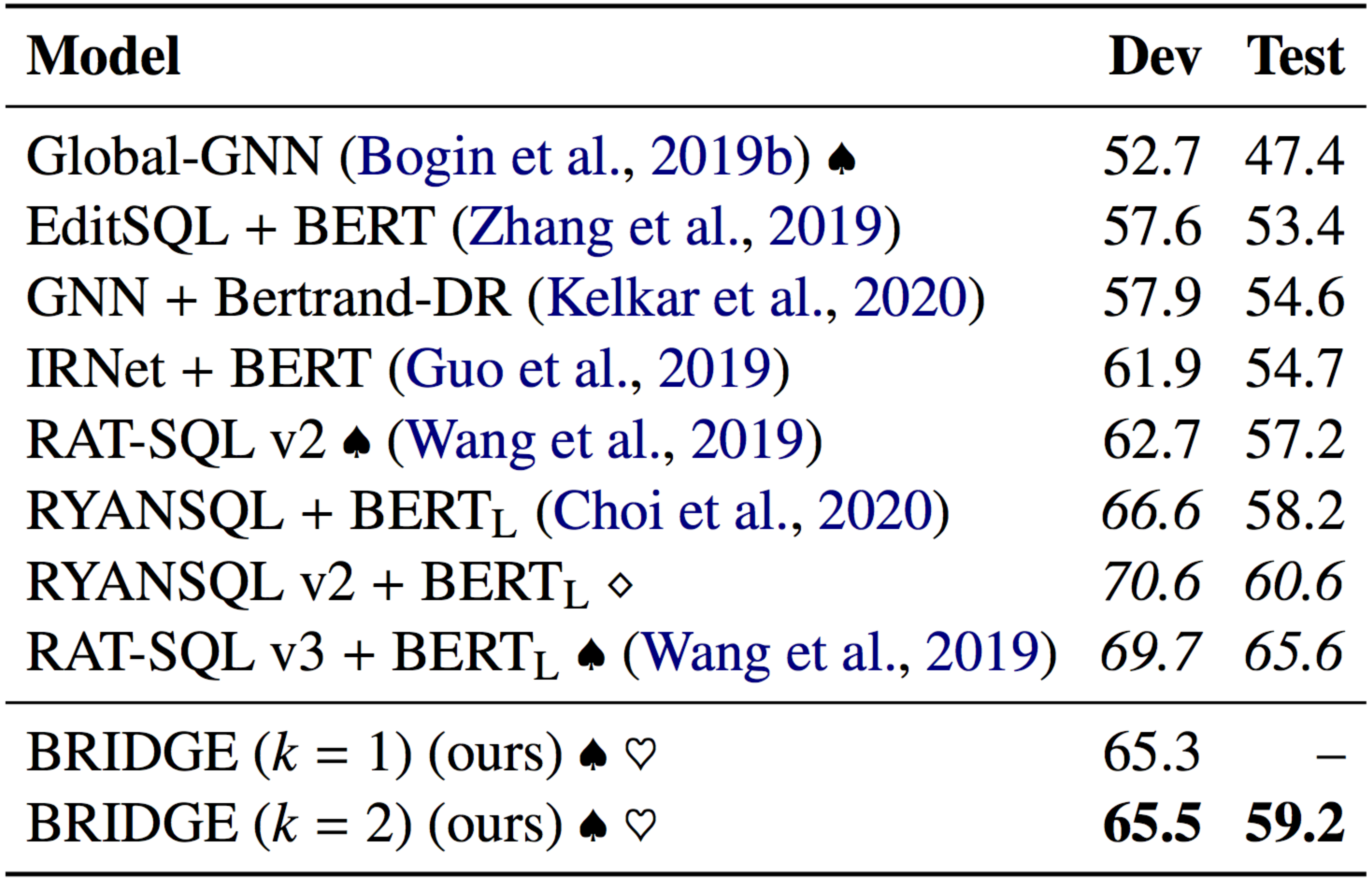


Model	**Exact Set Match (%)**	
	Mean	Max
BRIDGE (k = 2)	65.8 ± 0.8	66.9
- SC-guided decoding	65.4 ± 0.7	66.3 **(-0.6)**
- static SQL check	64.8 ± 0.9	65.9 **(-1.0)**
- execution order	64.2 ± 0.1	64.3 **(-2.6)**
- table shuffle & drop	63.9 ± 0.3	64.3 **(-2.6)**
- anchor text	63.3 ± 0.6	63.9 **(-3.0)**
- BERT	17.7 ± 0.7	18.3 **(-48.6)**



Model	**Easy**	**Medium**	**Hard**	**Ex-Hard**	**All**
count	250	440	174	170	1034
BRIDGE (k = 2)	**88.7**	**68.4**	**54**	**44**	**66.9**
-value augmentation	85.5	66.6	49.4	39.8	63.9

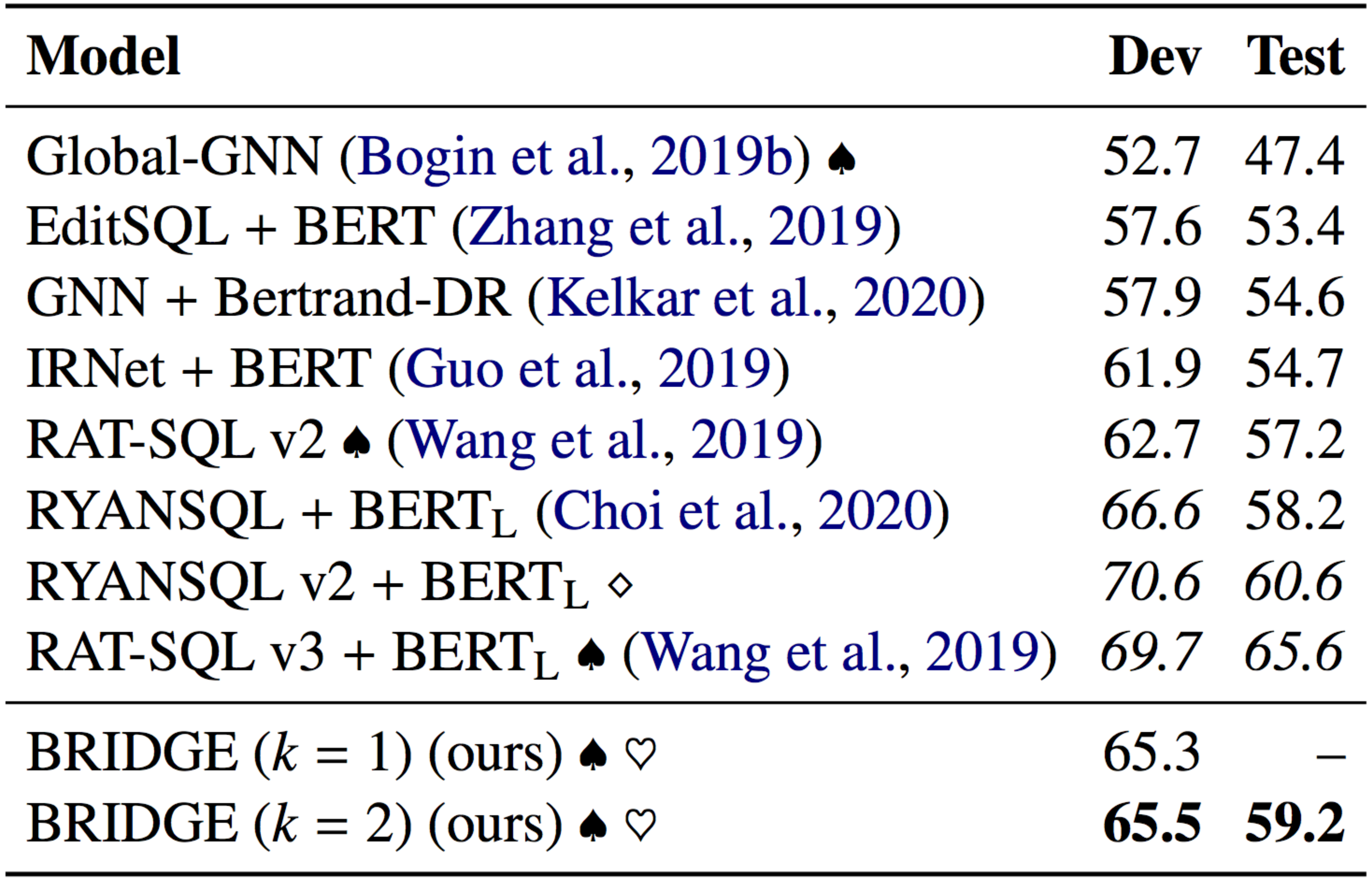
Leaderboard Performance



Model	Dev	Test
Global-GNN (Bogin et al., 2019b) ♠	52.7	47.4
EditSQL + BERT (Zhang et al., 2019)	57.6	53.4
GNN + Bertrand-DR (Kelkar et al., 2020)	57.9	54.6
IRNet + BERT (Guo et al., 2019)	61.9	54.7
RAT-SQL v2 ♠ (Wang et al., 2019)	62.7	57.2
RYANSQL + BERT$_{\text{L}}$ (Choi et al., 2020)	*66.6*	58.2
RYANSQL v2 + BERT$_{\text{L}}$ ◇	*70.6*	*60.6*
RAT-SQL v3 + BERT$_{\text{L}}$ ♠ (Wang et al., 2019)	*69.7*	*65.6*
BRIDGE (k = 1) (ours) ♠ ♡	65.3	–
BRIDGE (k = 2) (ours) ♠ ♡	**65.5**	**59.2**

(Spider leaderboard as of June 1st, 2020)

Leaderboard Performance



Model	Dev	Test
Global-GNN (Bogin et al., 2019b) ♠	52.7	47.4
EditSQL + BERT (Zhang et al., 2019)	57.6	53.4
GNN + Bertrand-DR (Kelkar et al., 2020)	57.9	54.6
IRNet + BERT (Guo et al., 2019)	61.9	54.7
RAT-SQL v2 ♠ (Wang et al., 2019)	62.7	57.2
RYANSQL + BERT$_L$ (Choi et al., 2020)	*66.6*	58.2
RYANSQL v2 + BERT$_L$ ◇	*70.6*	*60.6*
RAT-SQL v3 + BERT$_L$ ♠ (Wang et al., 2019)	*69.7*	*65.6*
BRIDGE (k = 1) (ours) ♠ ♡	65.3	–
BRIDGE (k = 2) (ours) ♠ ♡	**65.5**	**59.2**

(Spider leaderboard as of June 1st, 2020)

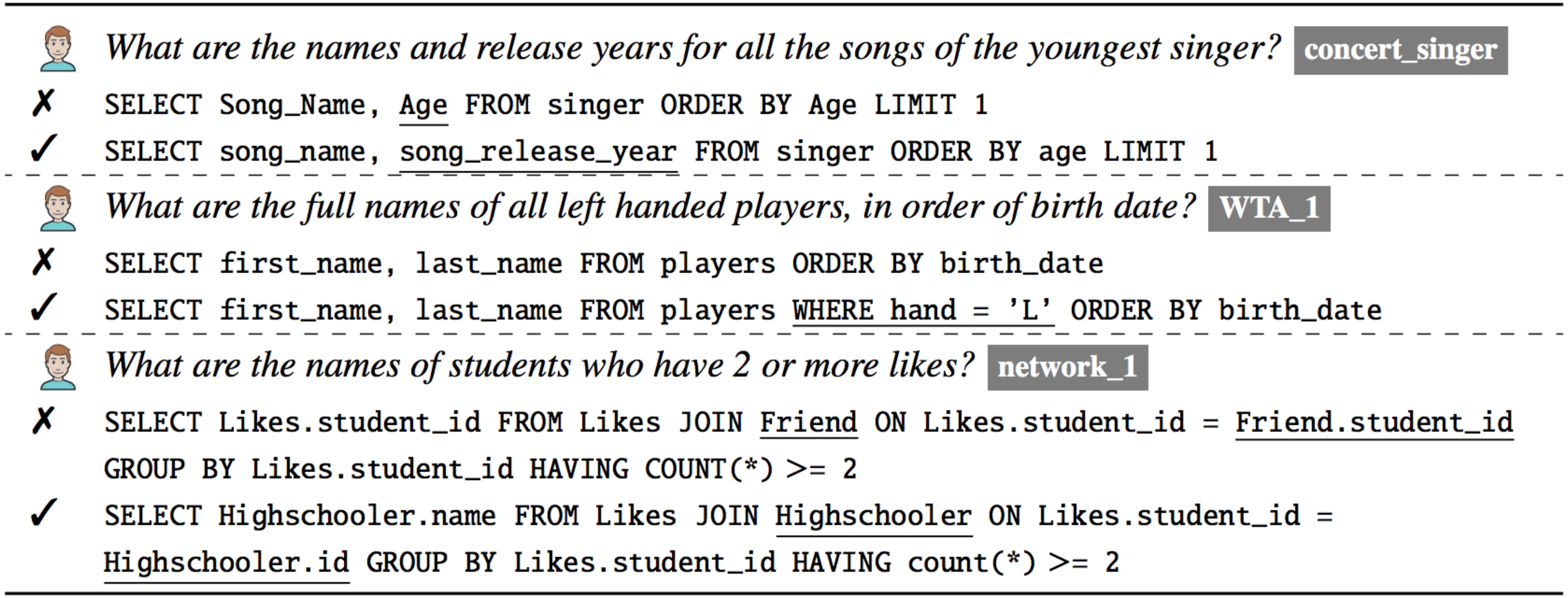
New results as of Nov. 20, 2020

With BERT-large: 70.0 (dev), 65.0 (test)

Our model synthesizes complete SQL queries

Error Analysis

Qualitative observations



What are the names and release years for all the songs of the youngest singer? **concert_singer**

✗ `SELECT Song_Name, Age FROM singer ORDER BY Age LIMIT 1`

✓ `SELECT song_name, song_release_year FROM singer ORDER BY age LIMIT 1`

What are the full names of all left handed players, in order of birth date? **WTA_1**

✗ `SELECT first_name, last_name FROM players ORDER BY birth_date`

✓ `SELECT first_name, last_name FROM players WHERE hand = 'L' ORDER BY birth_date`

What are the names of students who have 2 or more likes? **network_1**

✗ `SELECT Likes.student_id FROM Likes JOIN Friend ON Likes.student_id = Friend.student_id GROUP BY Likes.student_id HAVING COUNT(*) >= 2`

✓ `SELECT Highschooler.name FROM Likes JOIN Highschooler ON Likes.student_id = Highschooler.id GROUP BY Likes.student_id HAVING count(*) >= 2`

Robustness issue

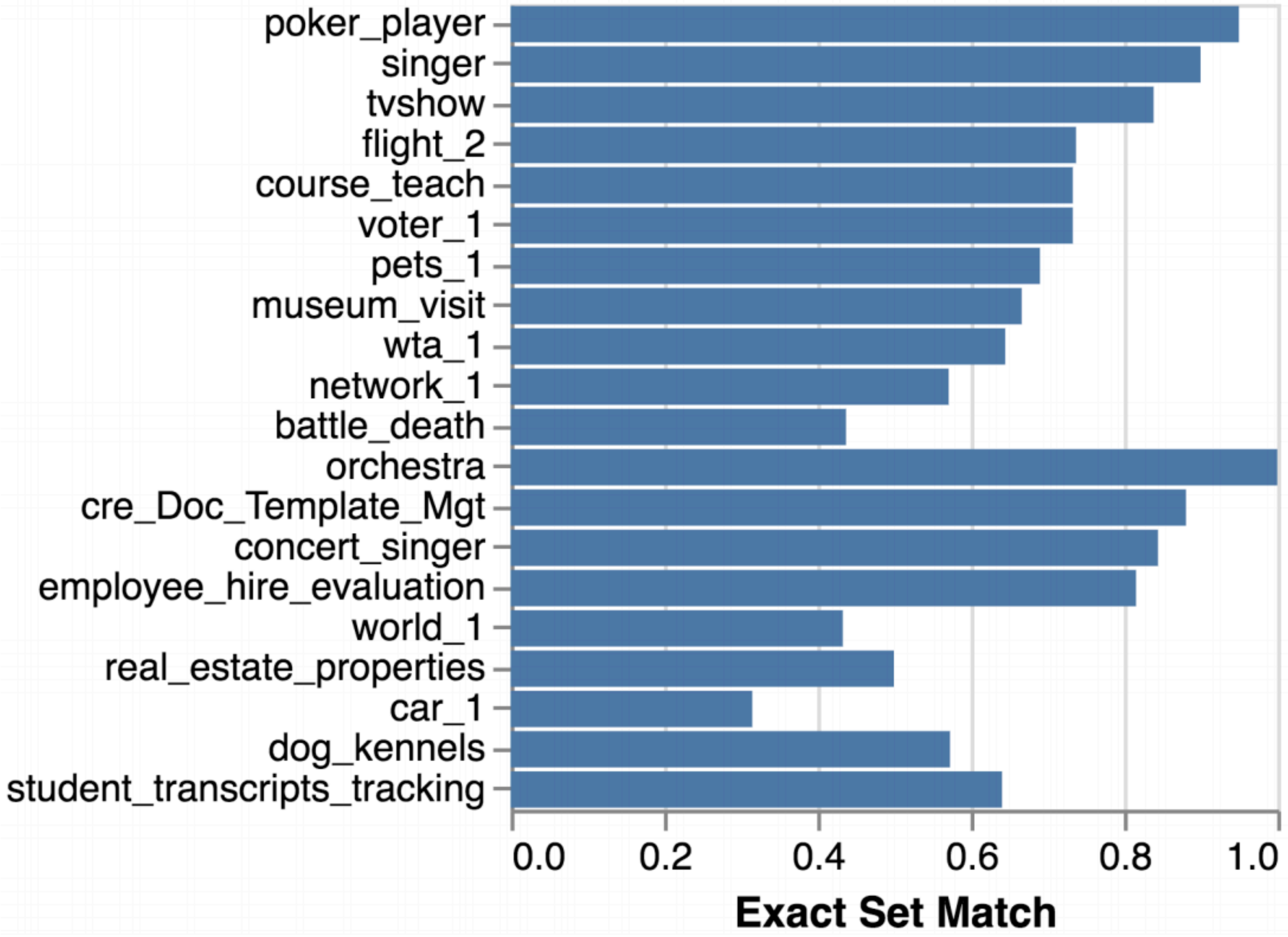
Rare relation & value surface form

Commonsense

“Friend” table stores students who has a friend, not all students

Performance by DB

Exact match accuracy on each DB in the Spider dev set. The DBs are sorted by size (smallest -> largest) from top to bottom.

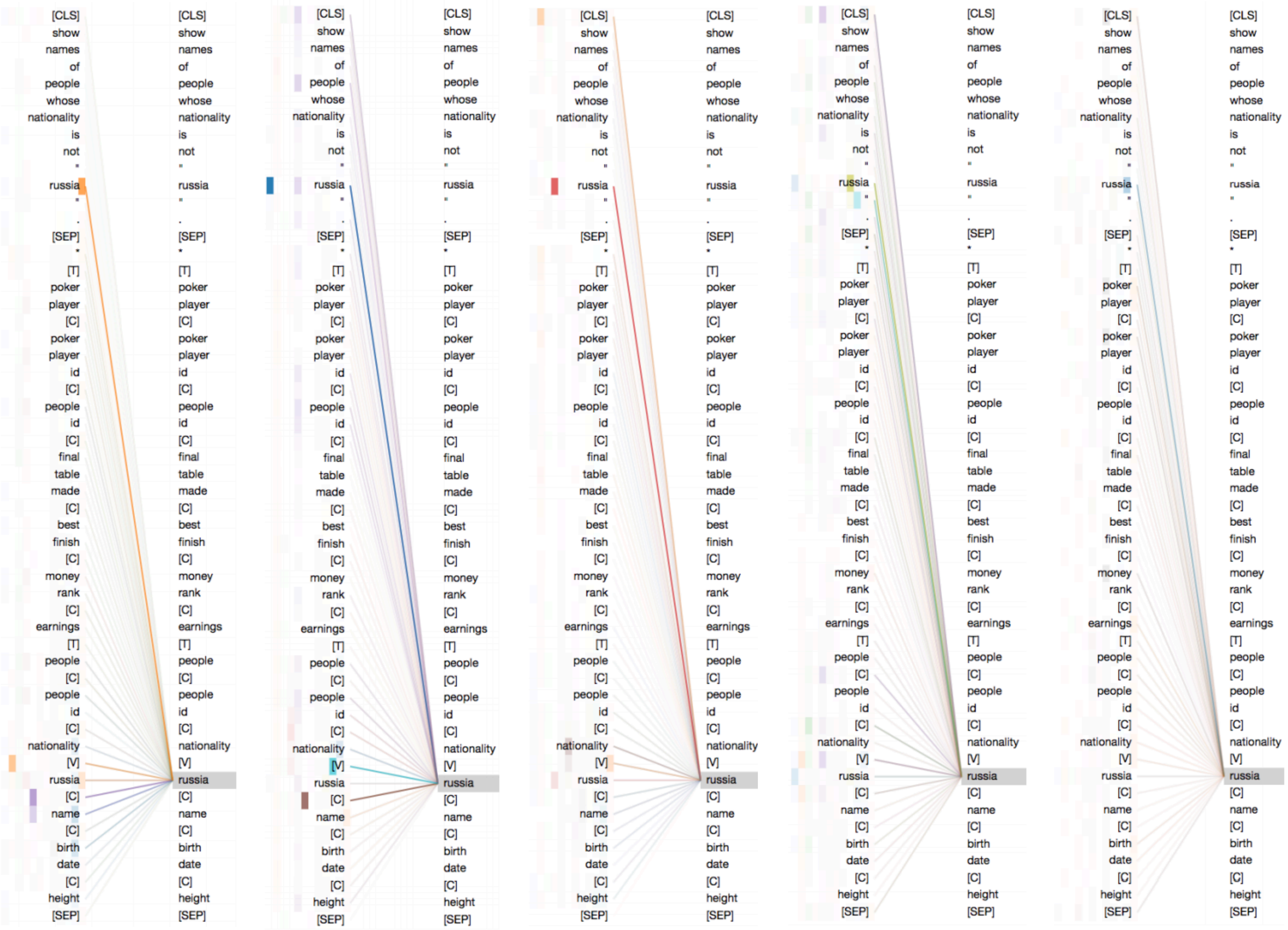


Better characterization of "similar" examples could help transfer learning

Fine-tuned BERT Attention Visualization

BertViz (Vig 2019)

Bridging

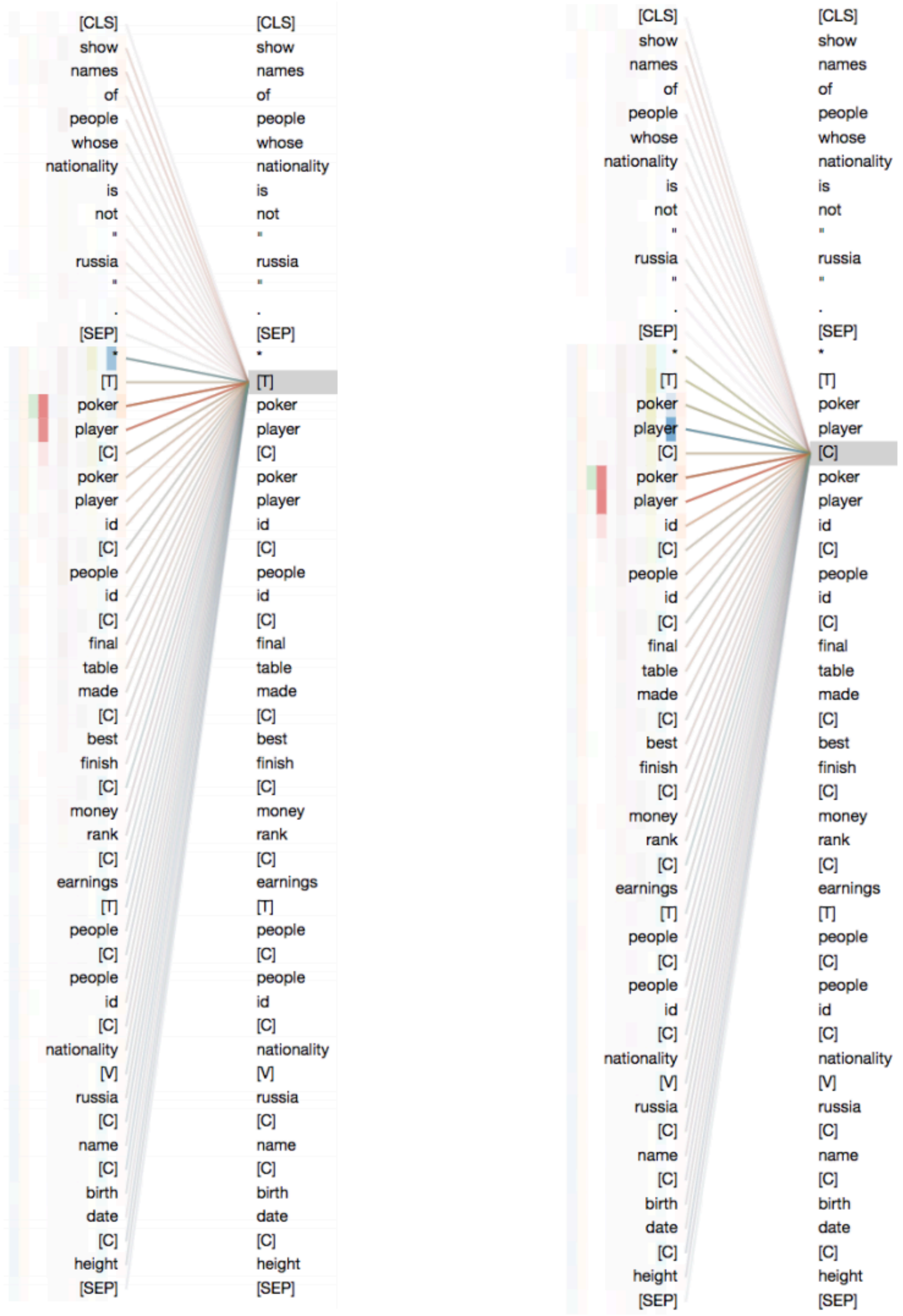


(a) Layer = 2 (b) Layer = 4 (c) Layer = 5 (a) Layer = 10 (b) Layer = 11

Fine-tuned BERT Attention Visualization

BertViz (Vig 2019)

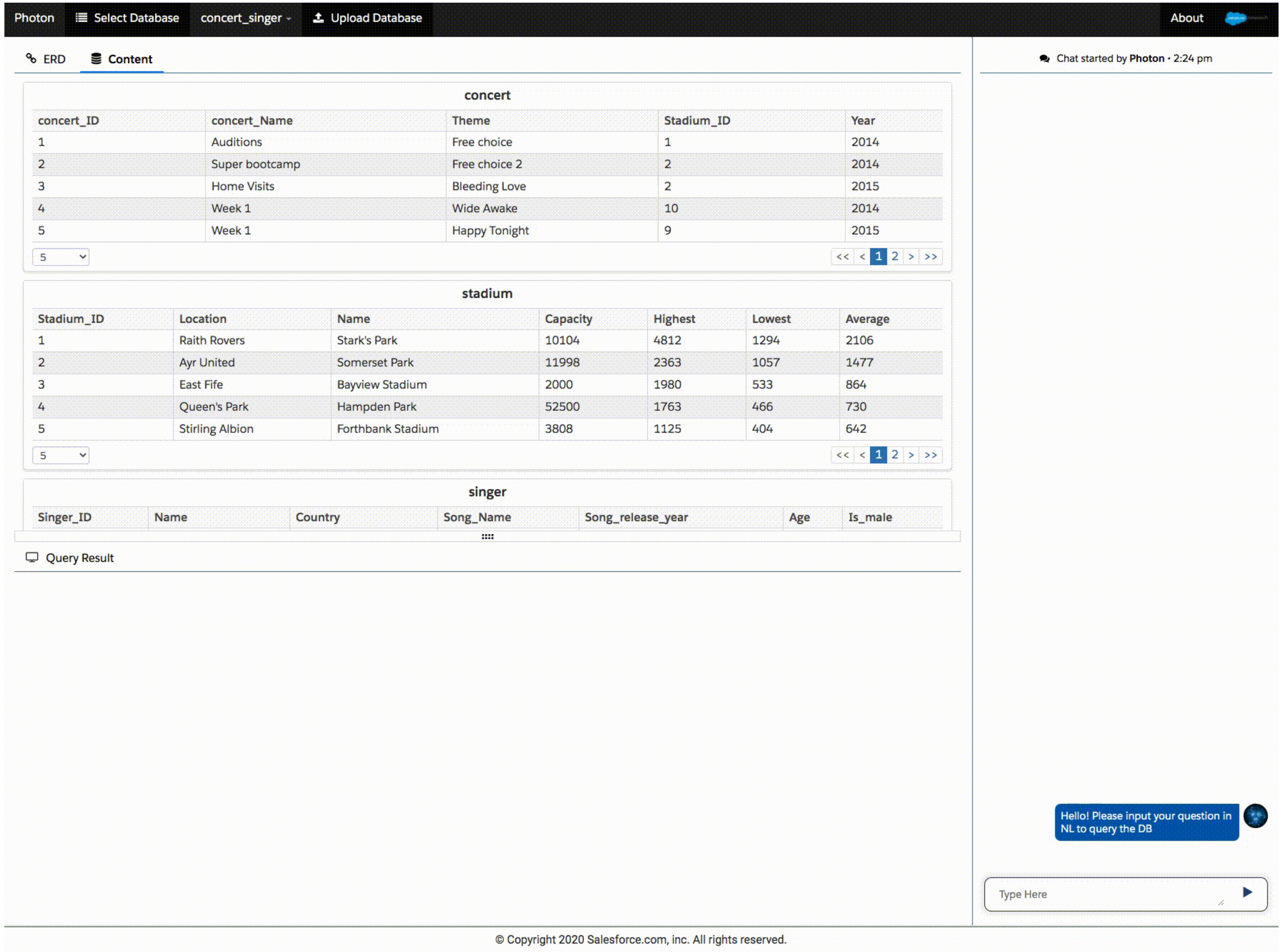
- Pooling effect in special tokens [T] and [C], layer 1



(a)

(b)

Live Demo https://naturalsql.com



Chat started by **Photon** • 2:24 pm

concert

concert_ID	concert_Name	Theme	Stadium_ID	Year
1	Auditions	Free choice	1	2014
2	Super bootcamp	Free choice 2	2	2014
3	Home Visits	Bleeding Love	2	2015
4	Week 1	Wide Awake	10	2014
5	Week 1	Happy Tonight	9	2015

5 << < 1 2 > >>

stadium

Stadium_ID	Location	Name	Capacity	Highest	Lowest	Average
1	Raith Rovers	Stark's Park	10104	4812	1294	2106
2	Ayr United	Somerset Park	11998	2363	1057	1477
3	East Fife	Bayview Stadium	2000	1980	533	864
4	Queen's Park	Hampden Park	52500	1763	466	730
5	Stirling Albion	Forthbank Stadium	3808	1125	404	642

5 << < 1 2 > >>

singer

Singer_ID	Name	Country	Song_Name	Song_release_year	Age	Is_male

Hello! Please input your question in NL to query the DB

© Copyright 2020 Salesforce.com, inc. All rights reserved.

Can be turned into a data collection tool

Takeaway

- Contextualizing the input utterance, DB schema structure and DB content is critical for text-to-SQL semantic parsing.

Takeaway

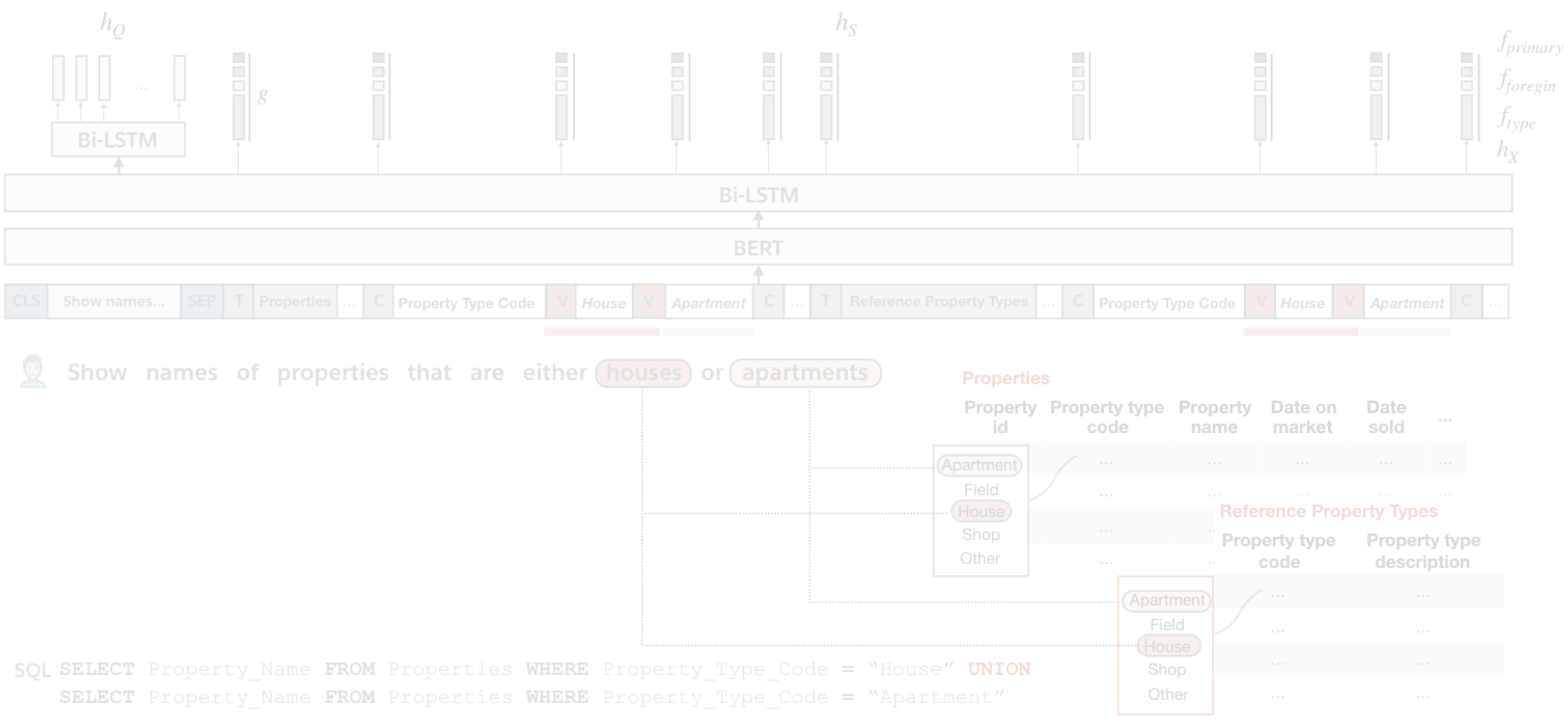
- Contextualizing the input utterance, DB schema structure and DB content is critical for text-to-SQL semantic parsing.
- By stretching the usage of special tokens in pre-trained language models we can effectively model such contextualization using multi-head self-attention over tagged sequences.

Takeaway

- Contextualizing the input utterance, DB schema structure and DB content is critical for text-to-SQL semantic parsing.
- By stretching the usage of special tokens in pre-trained language models we can effectively model such contextualization using multi-head self-attention over tagged sequences.
- Explicitly modeling the “structures” of data could still offer benefit and is worth exploring.
- Trustworthiness, interpretation and robustness are all critical for practical text-to-SQL semantic parser deployment.

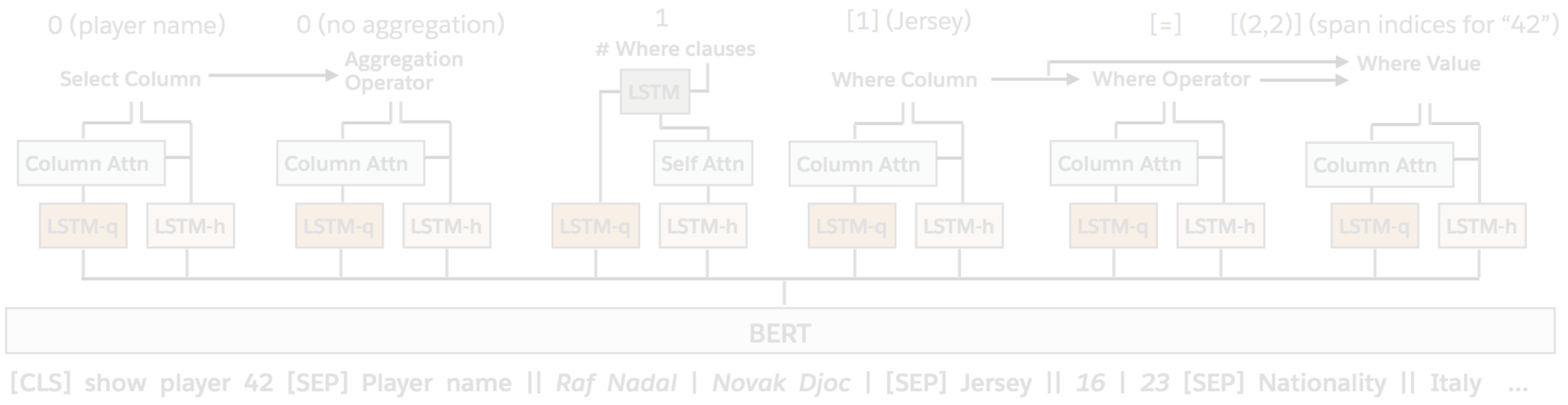
I. Content-Aware Textual-Tabular Encodings for Table Semantic Parsing (TSP)

Bridging Textual and Tabular Data for Cross-Domain Text-to-SQL Semantic Parsing. Lin et al. 2020.



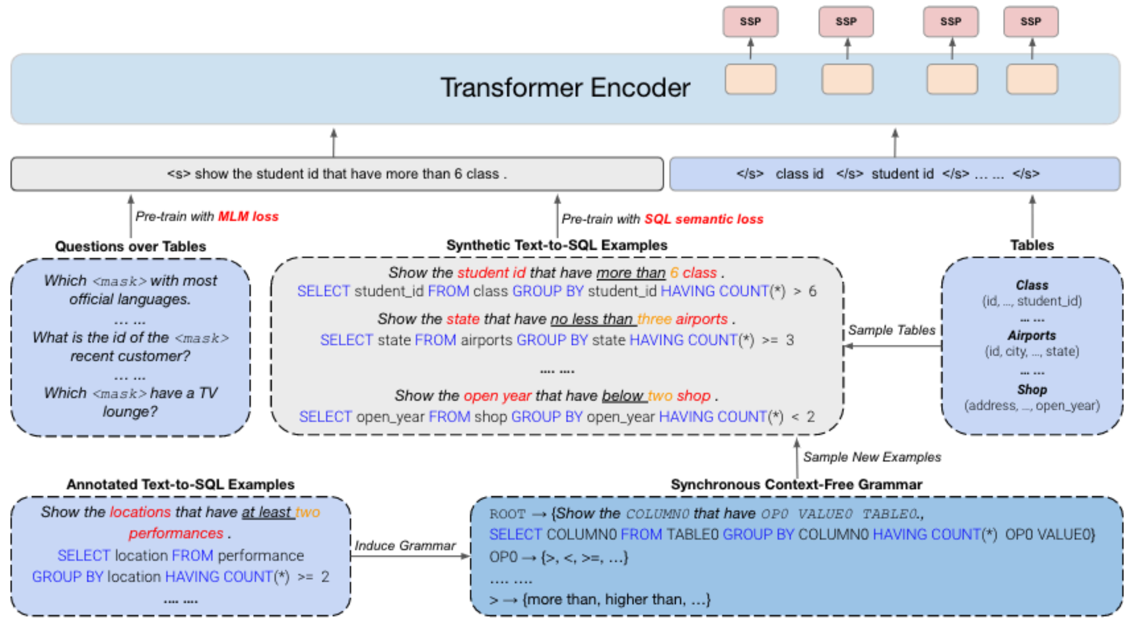
```
SQL SELECT Property_Name FROM Properties WHERE Property_Type_Code = "House" UNION
    SELECT Property_Name FROM Properties WHERE Property_Type_Code = "Apartment"
```

ColloQL: Robust Cross-Domain Text-to-SQL over Search Queries. Radhakrishnan et al. 2020.



II. Pre-training Textual-Tabular Representations with Semantic Scaffolds

GraPPa: Grammar-Augmented Pre-training for Table Semantic Parsing. Yu et al. 2020.



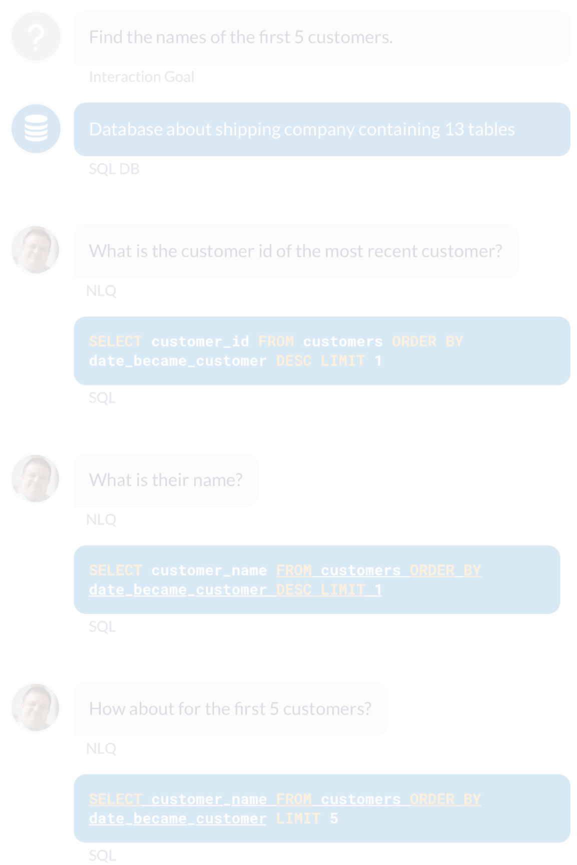
SSP
SSP
SSP
SSP
Transformer Encoder
<s> show the student id that have more than 6 class .
</s> class id </s> student id </s> ... ... </s>
Pre-train with MLM loss
Pre-train with SQL semantic loss
Questions over Tables
Which <mask> with most official languages.
... ...
What is the id of the <mask> recent customer?
... ...
Which <mask> have a TV lounge?
Synthetic Text-to-SQL Examples
Show the student id that have more than 6 class .
SELECT student_id FROM class GROUP BY student_id HAVING COUNT(*) > 6
Show the state that have no less than three airports .
SELECT state FROM airports GROUP BY state HAVING COUNT(*) >= 3
....
Show the open year that have below two shop .
SELECT open_year FROM shop GROUP BY open_year HAVING COUNT(*) < 2
Sample Tables
Tables
Class
(id, ..., student_id)
... ...
Airports
(id, city, ..., state)
... ...
Shop
(address, ..., open_year)
Sample New Examples
Annotated Text-to-SQL Examples
Show the locations that have at least two performances .
SELECT location FROM performance GROUP BY location HAVING COUNT(*) >= 2
....
Induce Grammar
Synchronous Context-Free Grammar
ROOT → {Show the COLUMN0 that have OP0 VALUE0 TABLE0.,
SELECT COLUMN0 FROM TABLE0 GROUP BY COLUMN0 HAVING COUNT(*) OP0 VALUE0}
OP0 → {>, <, >=, ...}
....
> → {more than, higher than, ...}

III. Conversational Table Semantic Parsing

SParC: Cross-Domain Semantic Parsing in Context. Yu et al. 2019.

Editing-Based SQL Query Generation for Cross-Domain Context-Dependent Questions. Zhang et al. 2019.

CoSQL: A Conversational Text-to-SQL Challenge Towards Cross-Domain Natural Language Interfaces to Databases. Yu et al. 2019.



Find the names of the first 5 customers.

Interaction Goal

Database about shipping company containing 13 tables

SQL DB

What is the customer id of the most recent customer?

NLQ

```
SELECT customer_id FROM customers ORDER BY date_became_customer DESC LIMIT 1
```

SQL

What is their name?

NLQ

```
SELECT customer_name FROM customers ORDER BY date_became_customer DESC LIMIT 1
```

SQL

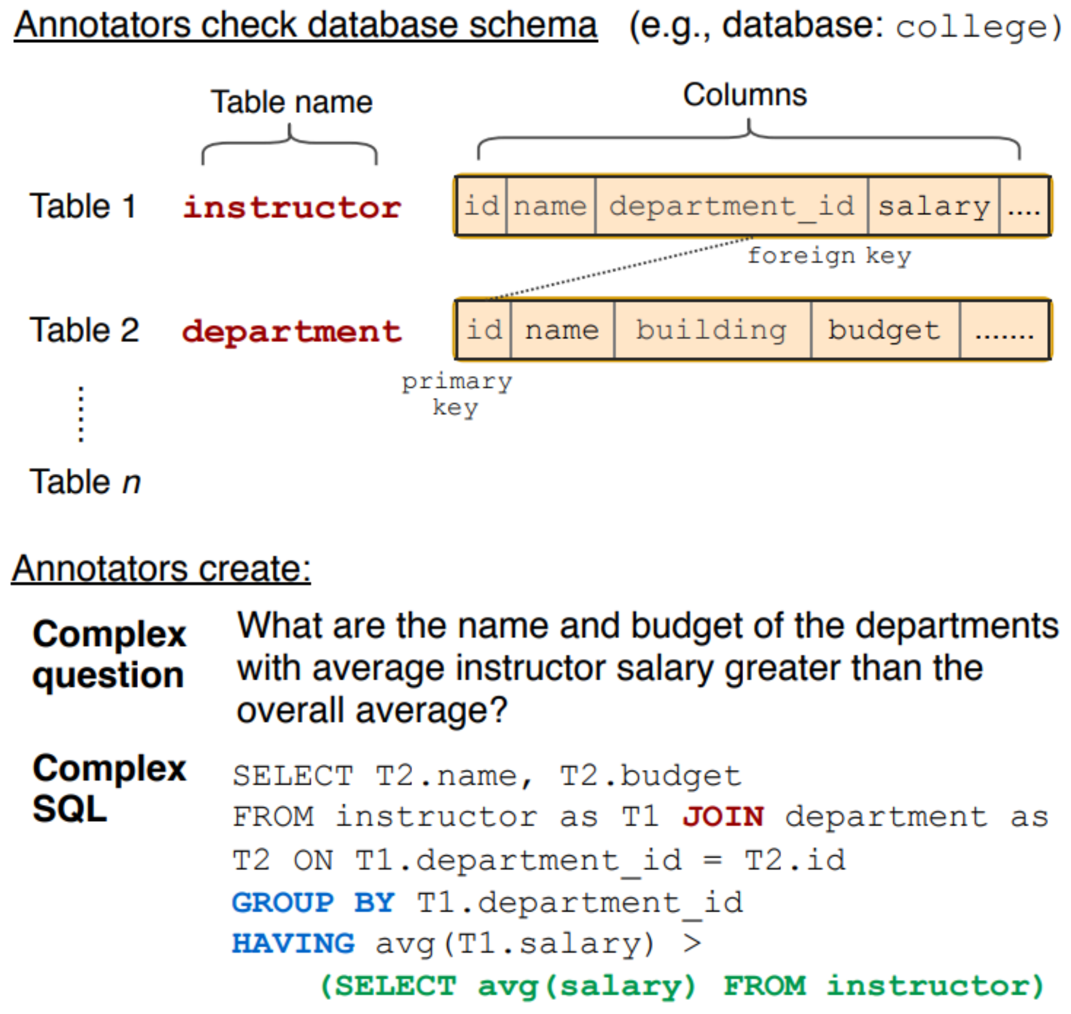
How about for the first 5 customers?

NLQ

```
SELECT customer_name FROM customers ORDER BY date_became_customer LIMIT 5
```

SQL

Language and Table Understanding



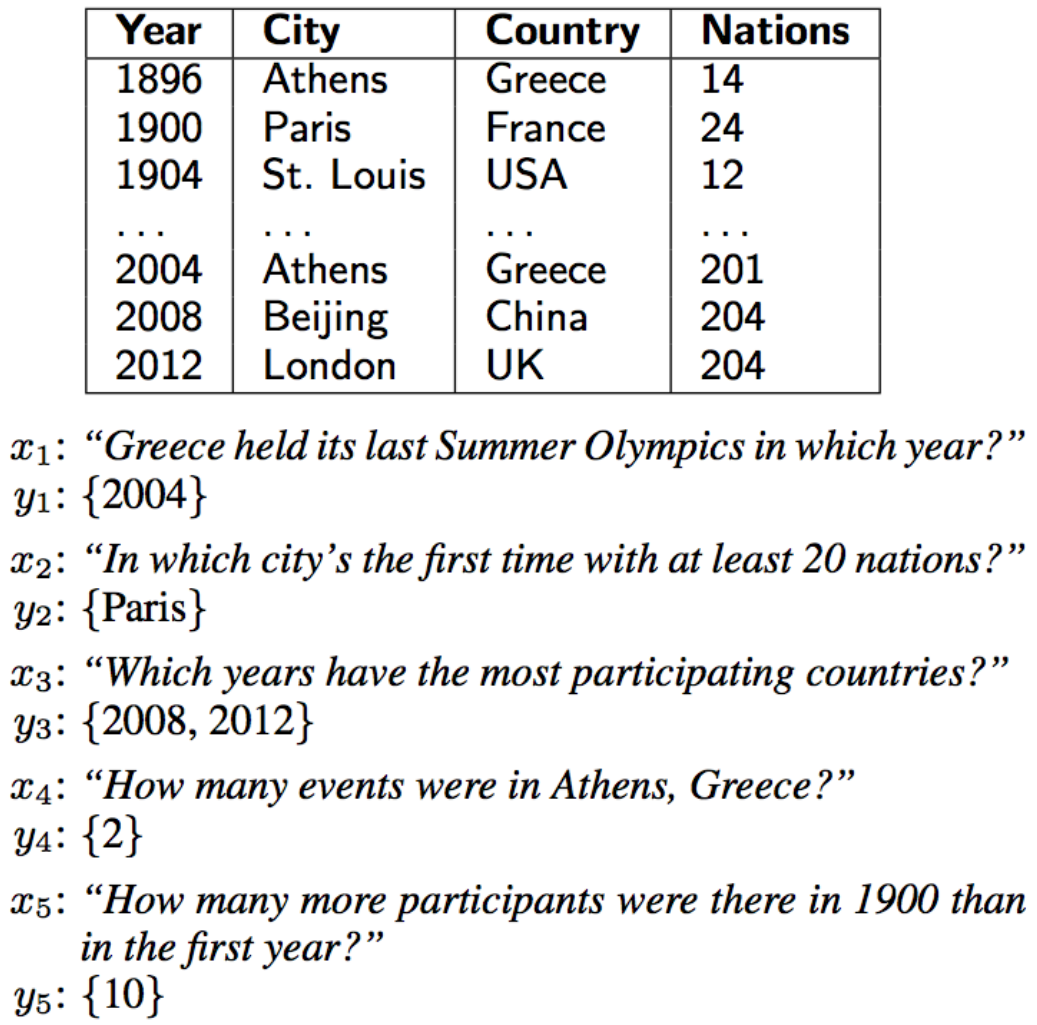
Annotators create:

Complex question What are the name and budget of the departments with average instructor salary greater than the overall average?

Complex SQL

```
SELECT T2.name, T2.budget
FROM instructor as T1 JOIN department as
T2 ON T1.department_id = T2.id
GROUP BY T1.department_id
HAVING avg(T1.salary) >
    (SELECT avg(salary) FROM instructor)
```

Semantic Parsing (Yu et al. 2020)



Year	City	Country	Nations
1896	Athens	Greece	14
1900	Paris	France	24
1904	St. Louis	USA	12
...	...	...	...
2004	Athens	Greece	201
2008	Beijing	China	204
2012	London	UK	204

x_1: *"Greece held its last Summer Olympics in which year?"*
y_1: {2004}

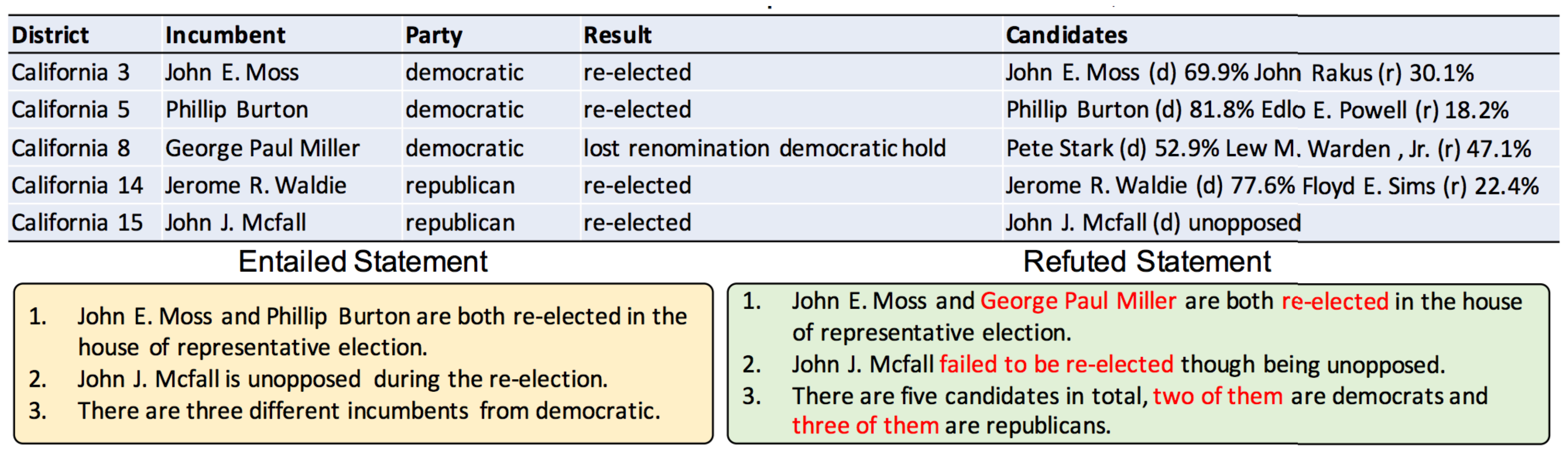
x_2: *"In which city's the first time with at least 20 nations?"*
y_2: {Paris}

x_3: *"Which years have the most participating countries?"*
y_3: {2008, 2012}

x_4: *"How many events were in Athens, Greece?"*
y_4: {2}

x_5: *"How many more participants were there in 1900 than in the first year?"*
y_5: {10}

Question Answering (Pasupat and Liang 2015)



District	Incumbent	Party	Result	Candidates
California 3	John E. Moss	democratic	re-elected	John E. Moss (d) 69.9% John Rakus (r) 30.1%
California 5	Phillip Burton	democratic	re-elected	Phillip Burton (d) 81.8% Edlo E. Powell (r) 18.2%
California 8	George Paul Miller	democratic	lost renomination democratic hold	Pete Stark (d) 52.9% Lew M. Warden , Jr. (r) 47.1%
California 14	Jerome R. Waldie	republican	re-elected	Jerome R. Waldie (d) 77.6% Floyd E. Sims (r) 22.4%
California 15	John J. Mcfall	republican	re-elected	John J. Mcfall (d) unopposed

Entailed Statement

1. John E. Moss and Phillip Burton are both re-elected in the house of representative election.
2. John J. Mcfall is unopposed during the re-election.
3. There are three different incumbents from democratic.

Refuted Statement

1. John E. Moss and George Paul Miller are both re-elected in the house of representative election.
2. John J. Mcfall failed to be re-elected though being unopposed.
3. There are five candidates in total, two of them are democrats and three of them are republicans.

Fact Verification (Chen et al. 2020)

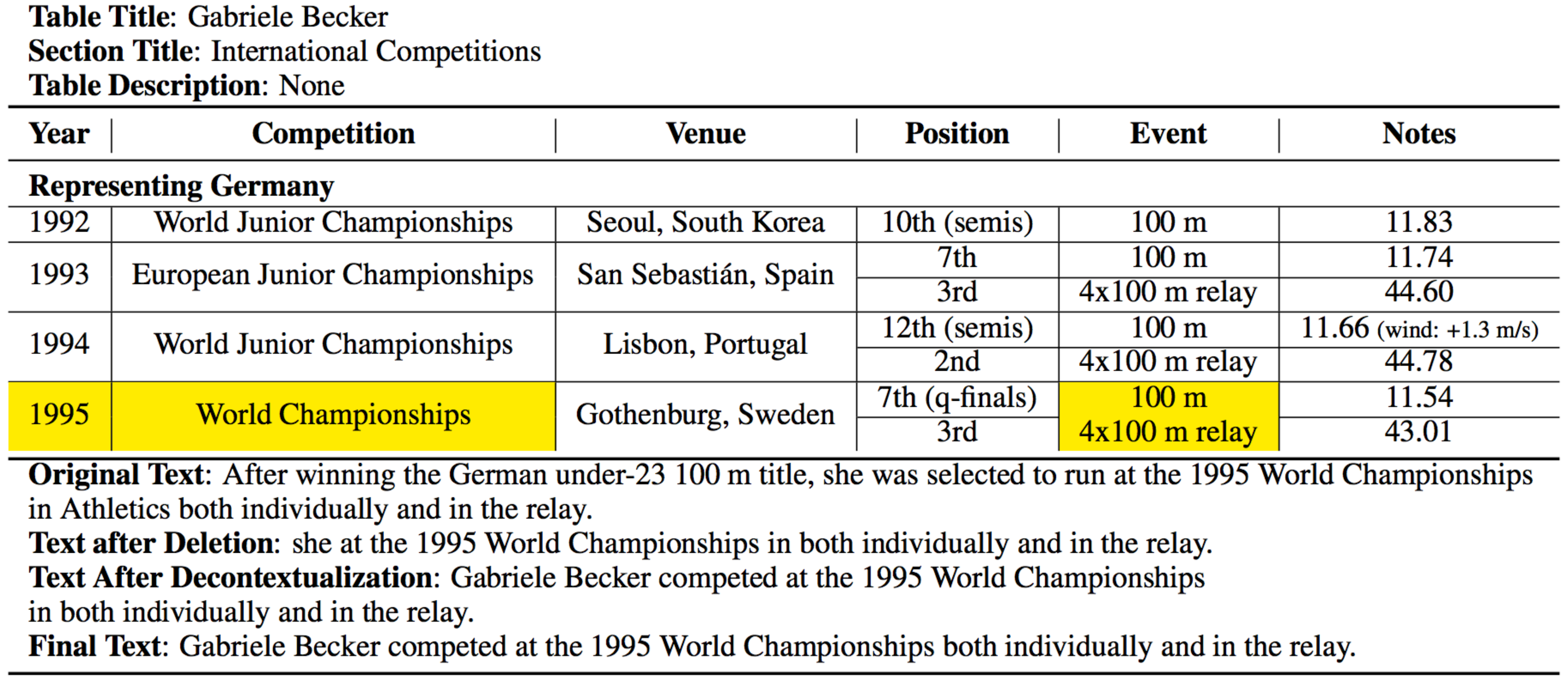


Table Title: Gabriele Becker
Section Title: International Competitions
Table Description: None

Year	Competition	Venue	Position	Event	Notes
Representing Germany					
1992	World Junior Championships	Seoul, South Korea	10th (semis)	100 m	11.83
1993	European Junior Championships	San Sebastián, Spain	7th	100 m	11.74
			3rd	4x100 m relay	44.60
1994	World Junior Championships	Lisbon, Portugal	12th (semis)	100 m	11.66 (wind: +1.3 m/s)
			2nd	4x100 m relay	44.78
1995	World Championships	Gothenburg, Sweden	7th (q-finals)	100 m	11.54
			3rd	4x100 m relay	43.01

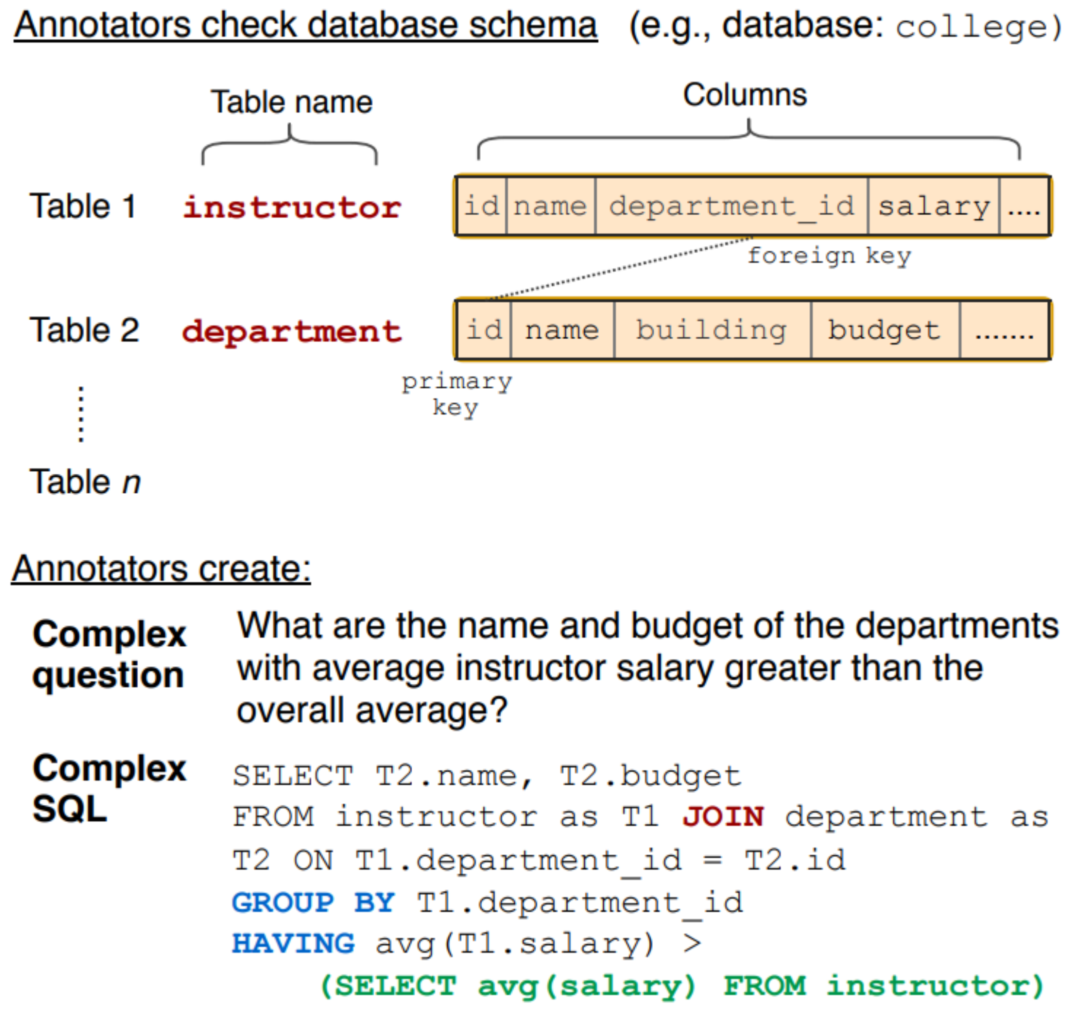
Original Text: After winning the German under-23 100 m title, she was selected to run at the 1995 World Championships in Athletics both individually and in the relay.
Text after Deletion: she at the 1995 World Championships in both individually and in the relay.
Text After Decontextualization: Gabriele Becker competed at the 1995 World Championships in both individually and in the relay.
Final Text: Gabriele Becker competed at the 1995 World Championships both individually and in the relay.

Table Summerization (Parikh et al. 2020)

There is growing need for table understanding in the field, often in the context of natural language...

GraPPa: Grammar-Augmented Pre-training for Table Semantic Parsing. Yu et al. 2020.

Language and Table Understanding



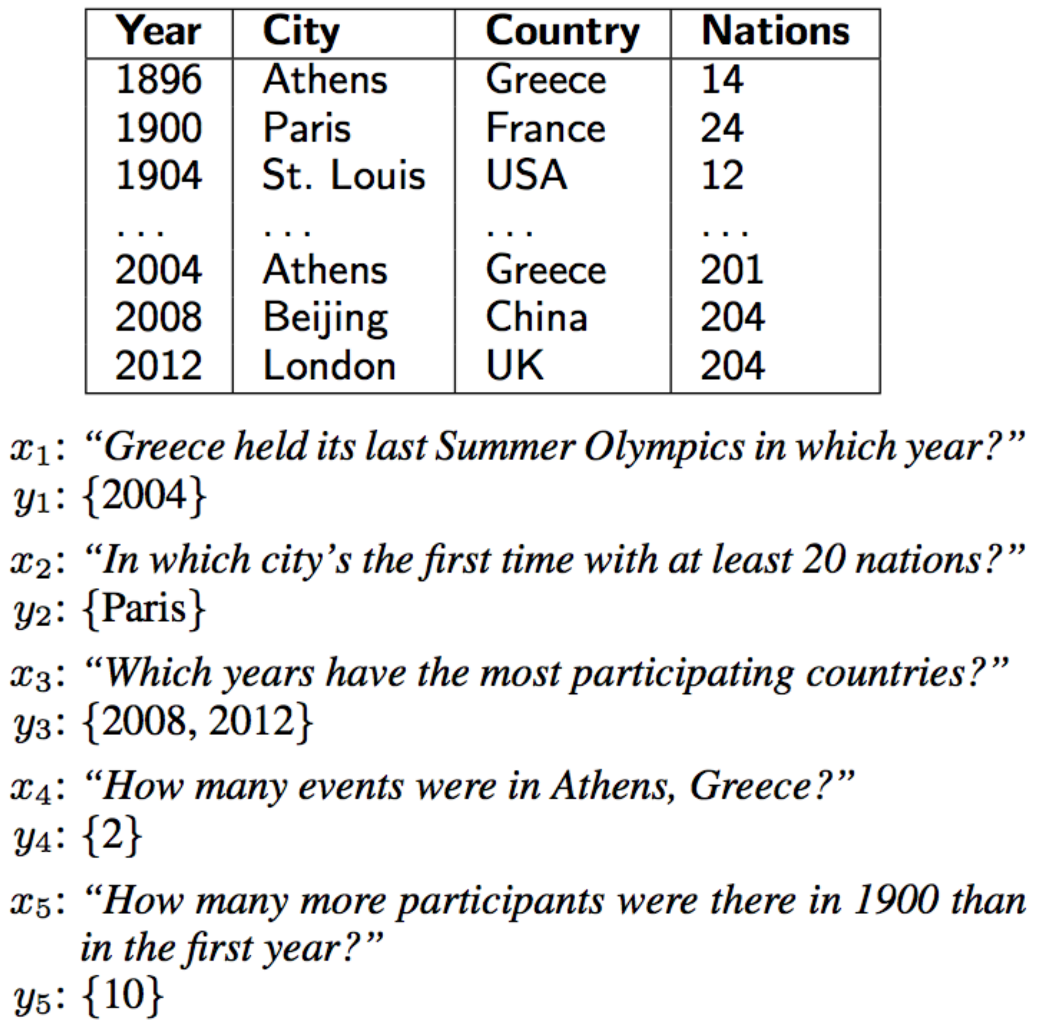
Annotators create:

Complex question What are the name and budget of the departments with average instructor salary greater than the overall average?

Complex SQL

```
SELECT T2.name, T2.budget
FROM instructor as T1 JOIN department as
T2 ON T1.department_id = T2.id
GROUP BY T1.department_id
HAVING avg(T1.salary) >
    (SELECT avg(salary) FROM instructor)
```

Semantic Parsing (Yu et al. 2020)



Year	City	Country	Nations
1896	Athens	Greece	14
1900	Paris	France	24
1904	St. Louis	USA	12
...	...	...	...
2004	Athens	Greece	201
2008	Beijing	China	204
2012	London	UK	204

x_1: *"Greece held its last Summer Olympics in which year?"*
y_1: {2004}

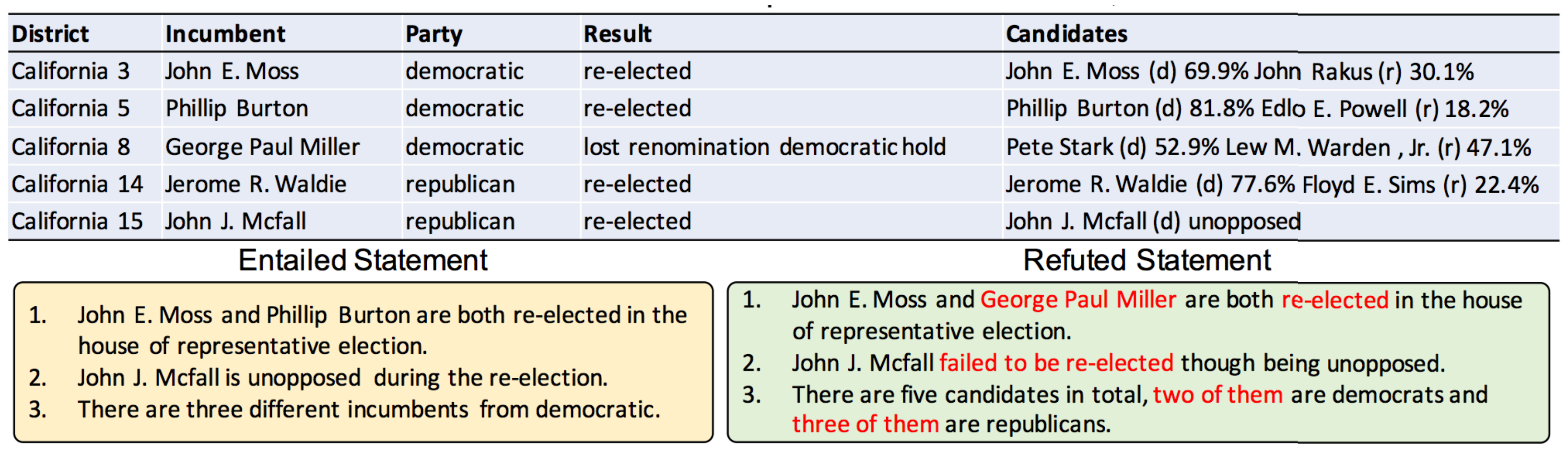
x_2: *"In which city's the first time with at least 20 nations?"*
y_2: {Paris}

x_3: *"Which years have the most participating countries?"*
y_3: {2008, 2012}

x_4: *"How many events were in Athens, Greece?"*
y_4: {2}

x_5: *"How many more participants were there in 1900 than in the first year?"*
y_5: {10}

Question Answering (Pasupat and Liang 2015)



District	Incumbent	Party	Result	Candidates
California 3	John E. Moss	democratic	re-elected	John E. Moss (d) 69.9% John Rakus (r) 30.1%
California 5	Phillip Burton	democratic	re-elected	Phillip Burton (d) 81.8% Edlo E. Powell (r) 18.2%
California 8	George Paul Miller	democratic	lost renomination democratic hold	Pete Stark (d) 52.9% Lew M. Warden , Jr. (r) 47.1%
California 14	Jerome R. Waldie	republican	re-elected	Jerome R. Waldie (d) 77.6% Floyd E. Sims (r) 22.4%
California 15	John J. Mcfall	republican	re-elected	John J. Mcfall (d) unopposed

Entailed Statement

1. John E. Moss and Phillip Burton are both re-elected in the house of representative election.
2. John J. Mcfall is unopposed during the re-election.
3. There are three different incumbents from democratic.

Refuted Statement

1. John E. Moss and George Paul Miller are both re-elected in the house of representative election.
2. John J. Mcfall failed to be re-elected though being unopposed.
3. There are five candidates in total, two of them are democrats and three of them are republicans.

Fact Verification (Chen et al. 2020)

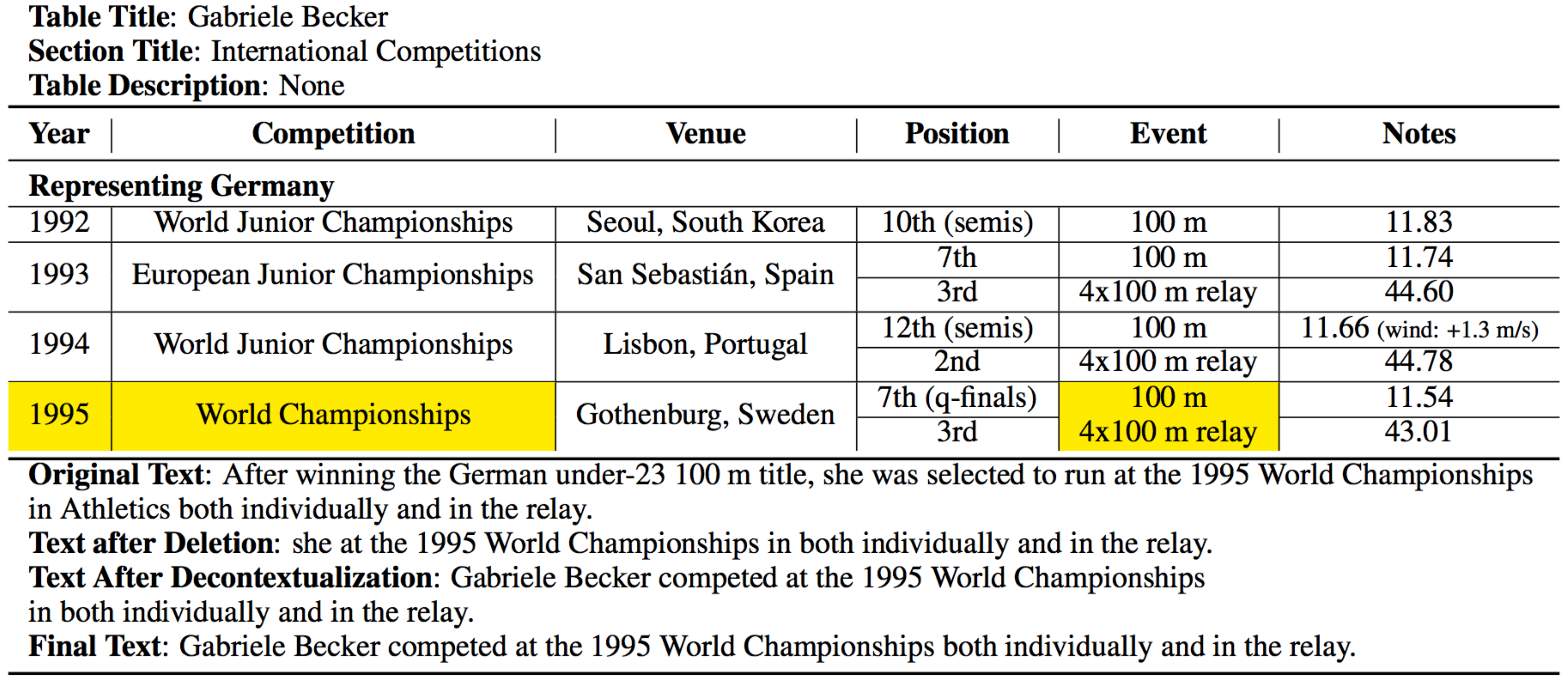


Table Title: Gabriele Becker
Section Title: International Competitions
Table Description: None

Year	Competition	Venue	Position	Event	Notes
Representing Germany					
1992	World Junior Championships	Seoul, South Korea	10th (semis)	100 m	11.83
1993	European Junior Championships	San Sebastián, Spain	7th	100 m	11.74
			3rd	4x100 m relay	44.60
1994	World Junior Championships	Lisbon, Portugal	12th (semis)	100 m	11.66 (wind: +1.3 m/s)
			2nd	4x100 m relay	44.78
1995	World Championships	Gothenburg, Sweden	7th (q-finals)	100 m	11.54
			3rd	4x100 m relay	43.01

Original Text: After winning the German under-23 100 m title, she was selected to run at the 1995 World Championships in Athletics both individually and in the relay.
Text after Deletion: she at the 1995 World Championships in both individually and in the relay.
Text After Decontextualization: Gabriele Becker competed at the 1995 World Championships in both individually and in the relay.
Final Text: Gabriele Becker competed at the 1995 World Championships both individually and in the relay.

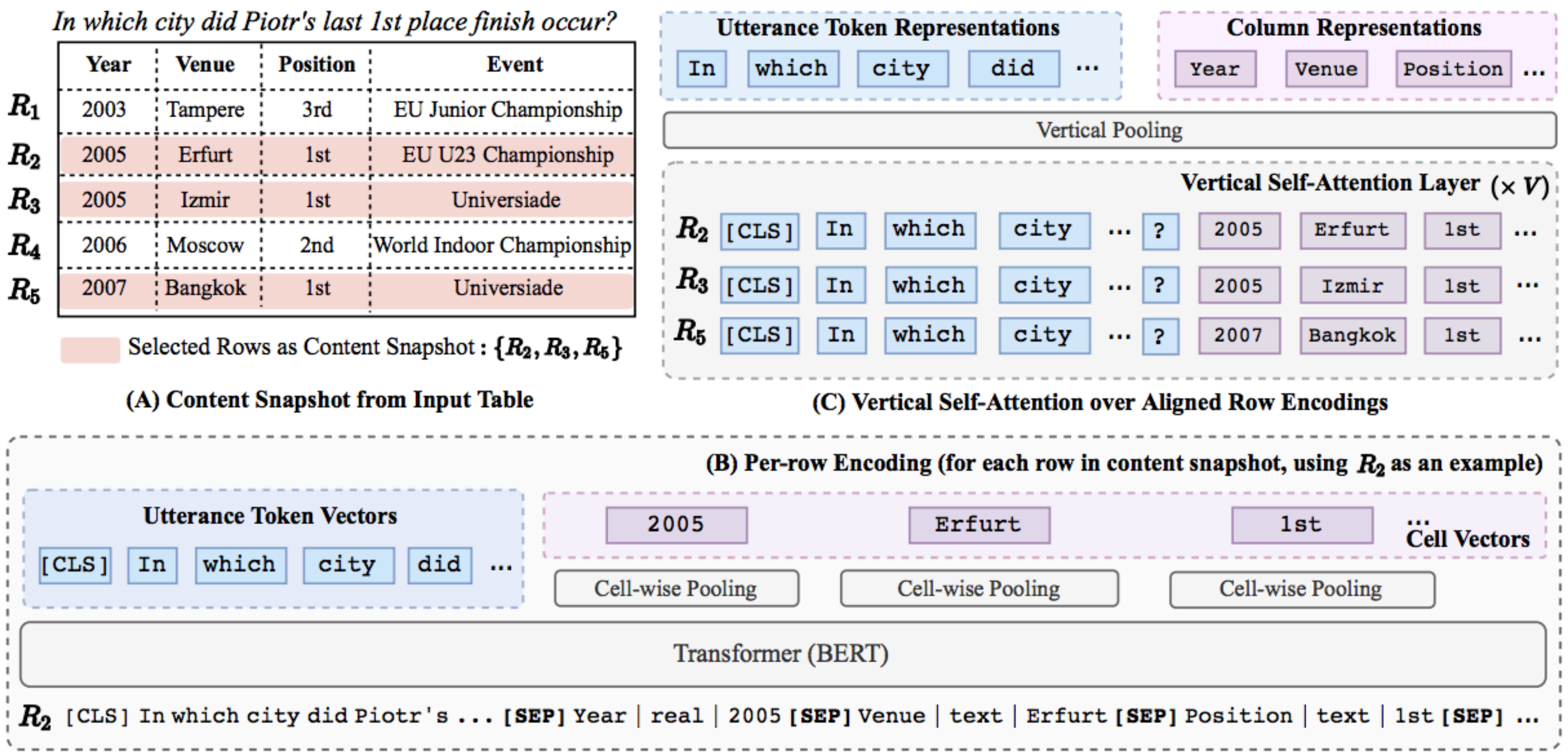
Table Summerization (Parikh et al. 2020)

We **pre-train joint representation for text and tables** with potential benefit across tasks, focusing on **table semantic parsing and question answering tasks.**

GraPPa: Grammar-Augmented Pre-training for Table Semantic Parsing. Yu et al. 2020.

Pre-trained Language and Table Representation

- ***Data:* 26M** *tables and their English contexts from* ***English Wikipedia*** *and* ***the WDC WebTable Corpus***
- ***Objective:*** *standard MLM; Masked Column Prediction (MCP); Cell Value Recovery (CVR)*
- ***Content Snapshot:*** *sampled rows that summarize the information in T most relevant to the input utterance*



In which city did Piotr's last 1st place finish occur?

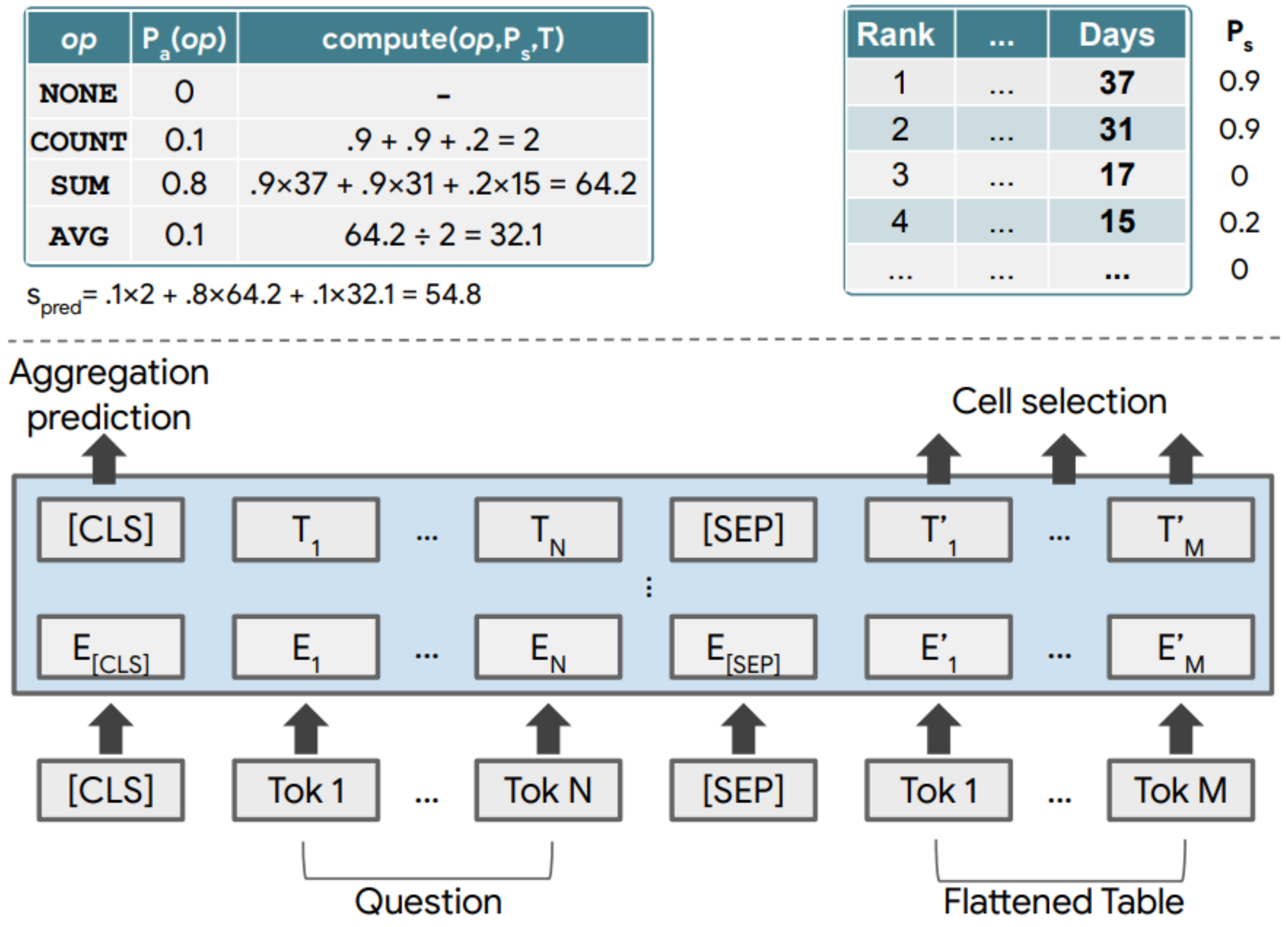
	Year	Venue	Position	Event
R_1	2003	Tampere	3rd	EU Junior Championship
R_2	2005	Erfurt	1st	EU U23 Championship
R_3	2005	Izmir	1st	Universiade
R_4	2006	Moscow	2nd	World Indoor Championship
R_5	2007	Bangkok	1st	Universiade

Selected Rows as Content Snapshot : $\{R_2, R_3, R_5\}$

(A) Content Snapshot from Input Table
Utterance Token Representations
In
which
city
did
...
Column Representations
Year
Venue
Position
...
Vertical Pooling
Vertical Self-Attention Layer (× V)
R2 [CLS] In which city ... ? 2005 Erfurt 1st ...
R3 [CLS] In which city ... ? 2005 Izmir 1st ...
R5 [CLS] In which city ... ? 2007 Bangkok 1st ...
(C) Vertical Self-Attention over Aligned Row Encodings
(B) Per-row Encoding (for each row in content snapshot, using R2 as an example)
Utterance Token Vectors
[CLS]
In
which
city
did
...
2005
Erfurt
1st
...
Cell Vectors
Cell-wise Pooling
Cell-wise Pooling
Cell-wise Pooling
Transformer (BERT)
R2 [CLS] In which city did Piotr's ... [SEP] Year | real | 2005 [SEP] Venue | text | Erfurt [SEP] Position | text | 1st [SEP] ...

TaBERT: Pretraining for Joint Understanding of Textual And Tabular Data (Yin et al. 2020)

- ***Data:* 3.3M Infoboxes** *and* **2.9M WikiTables** *with relevant text snippets including table caption, article title, article description, segment title and text of the segment*
- ***Objective:*** *standard MLM and relevant table prediction*
- ***Table Content*** *is flattened and inserted into the table schema*



op	$P_a(op)$	compute(op, P_s, T)
NONE	0	-
COUNT	0.1	.9 + .9 + .2 = 2
SUM	0.8	.9×37 + .9×31 + .2×15 = 64.2
AVG	0.1	64.2 ÷ 2 = 32.1

s_{pred} = .1×2 + .8×64.2 + .1×32.1 = 54.8

Rank	...	Days	P_s
1	...	37	0.9
2	...	31	0.9
3	...	17	0
4	...	15	0.2
...	...	...	0

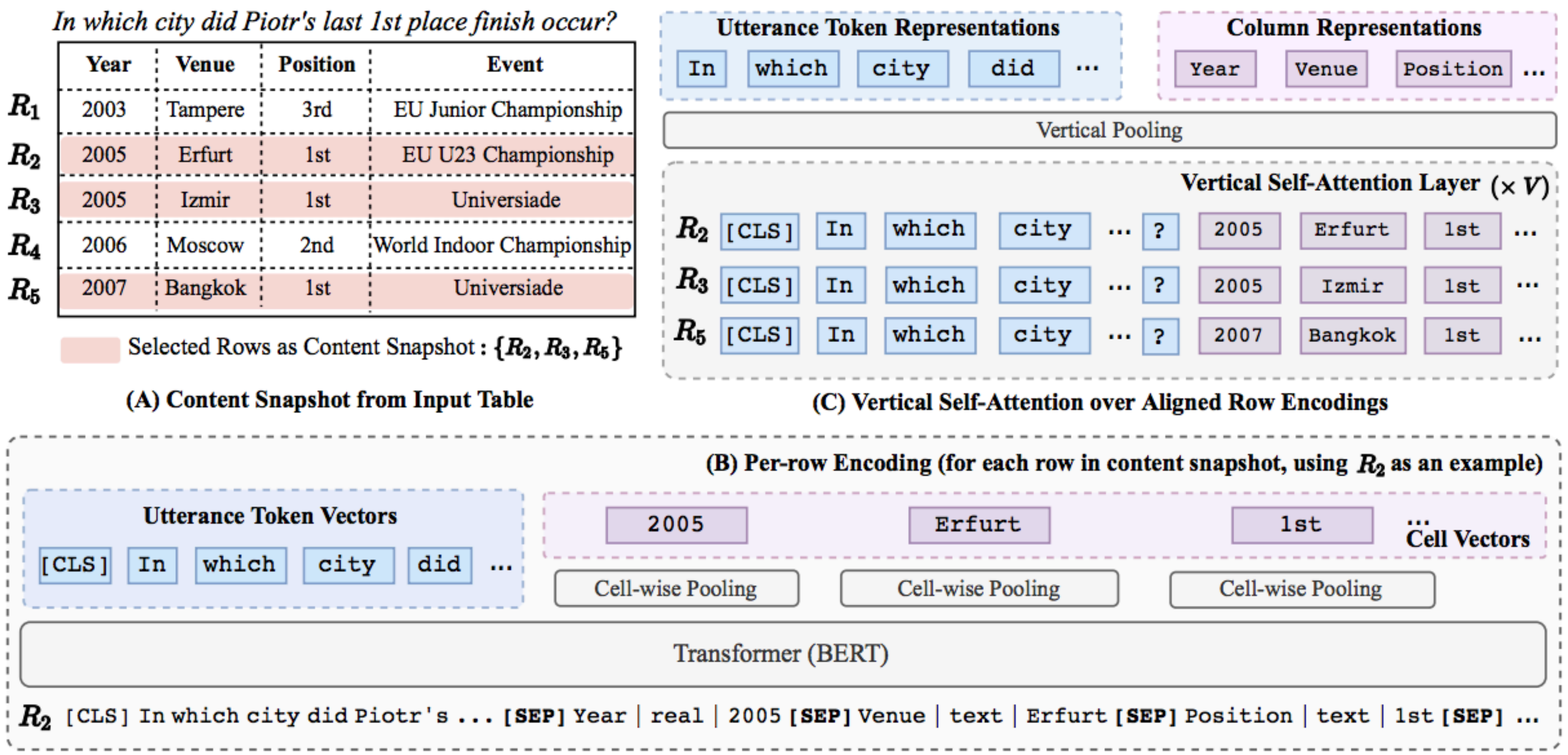
TAPAS: Weakly Supervised Table Parsing via Pre-training (Yin et al. 2020)

Challenges

- ***Data:* *26M* *tables and their English contexts from* *English Wikipedia* *and* *the WDC WebTable Corpus***

- ***Data: 3.3M Infoboxes*** *and* ***2.9M WikiTables*** *with relevant text snippets including table caption, article title, article description, segment title and text of the segment*

Large, noisy training data

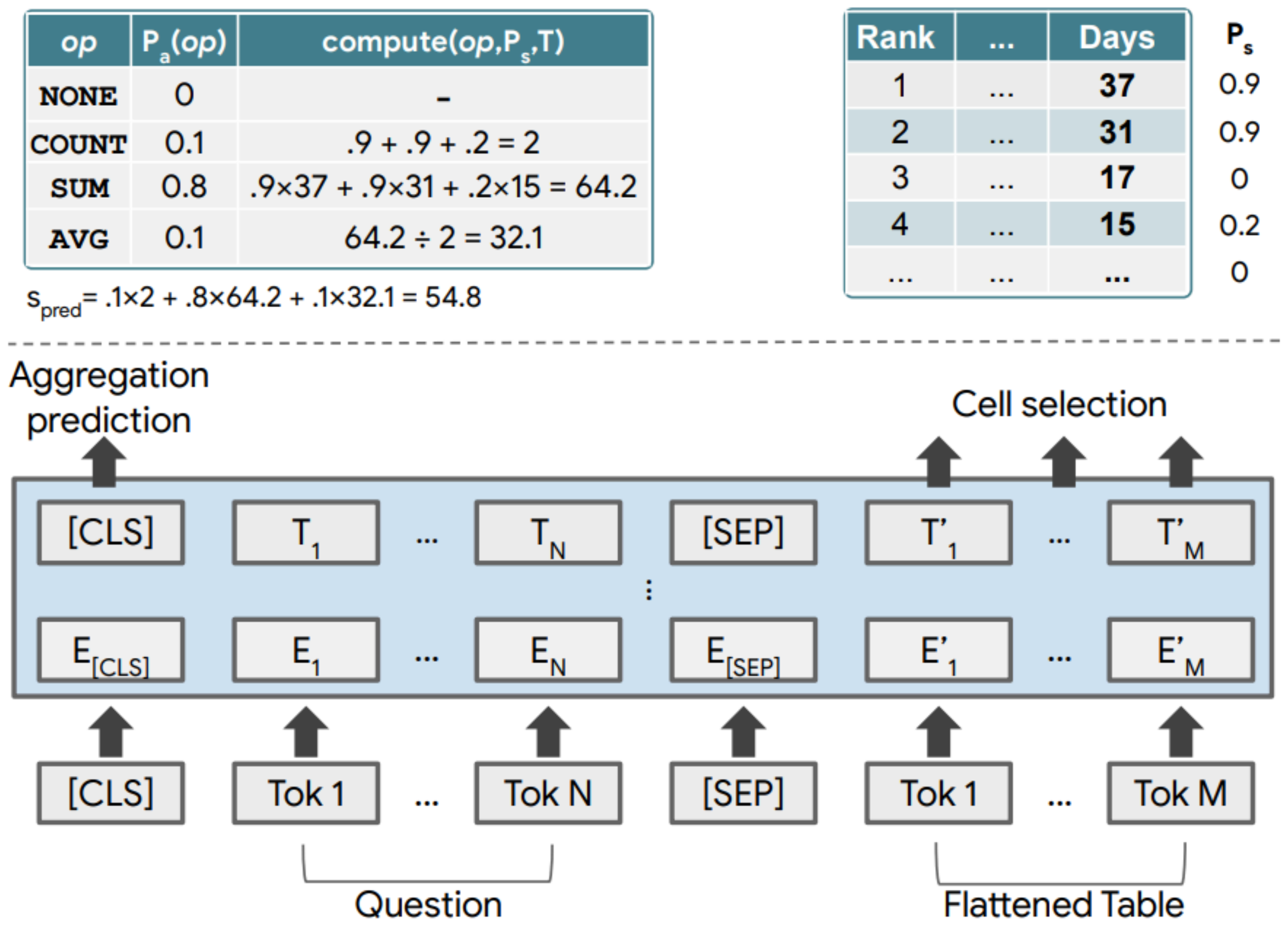


In which city did Piotr's last 1st place finish occur?

	Year	Venue	Position	Event
R_1	2003	Tampere	3rd	EU Junior Championship
R_2	2005	Erfurt	1st	EU U23 Championship
R_3	2005	Izmir	1st	Universiade
R_4	2006	Moscow	2nd	World Indoor Championship
R_5	2007	Bangkok	1st	Universiade

Selected Rows as Content Snapshot : $\{R_2, R_3, R_5\}$
(A) Content Snapshot from Input Table
Utterance Token Representations
In which city did ...
Column Representations
Year Venue Position ...
Vertical Pooling
Vertical Self-Attention Layer (× V)
R_2 [CLS] In which city ... ? 2005 Erfurt 1st ...
R_3 [CLS] In which city ... ? 2005 Izmir 1st ...
R_5 [CLS] In which city ... ? 2007 Bangkok 1st ...
(C) Vertical Self-Attention over Aligned Row Encodings
(B) Per-row Encoding (for each row in content snapshot, using R_2 as an example)
Utterance Token Vectors
[CLS] In which city did ...
2005 Erfurt 1st ...
Cell Vectors
Cell-wise Pooling
Cell-wise Pooling
Cell-wise Pooling
Transformer (BERT)
R_2 [CLS] In which city did Piotr's ... [SEP] Year | real | 2005 [SEP] Venue | text | Erfurt [SEP] Position | text | 1st [SEP] ...

TaBERT: Pretraining for Joint Understanding of Textual And Tabular Data (Yin et al. 2020)



op	$P_a(op)$	compute(op,P_s,T)
NONE	0	-
COUNT	0.1	.9 + .9 + .2 = 2
SUM	0.8	.9×37 + .9×31 + .2×15 = 64.2
AVG	0.1	64.2 ÷ 2 = 32.1

s_{pred}= .1×2 + .8×64.2 + .1×32.1 = 54.8

Rank	...	Days	P_s
1	...	37	0.9
2	...	31	0.9
3	...	17	0
4	...	15	0.2
...	...	...	0

Aggregation prediction
Cell selection
[CLS] T_1 ... T_N [SEP] T'_1 ... T'_M
$E_{[CLS]}$ E_1 ... E_N $E_{[SEP]}$ E'_1 ... E'_M
[CLS] Tok 1 ... Tok N [SEP] Tok 1 ... Tok M
Question
Flattened Table

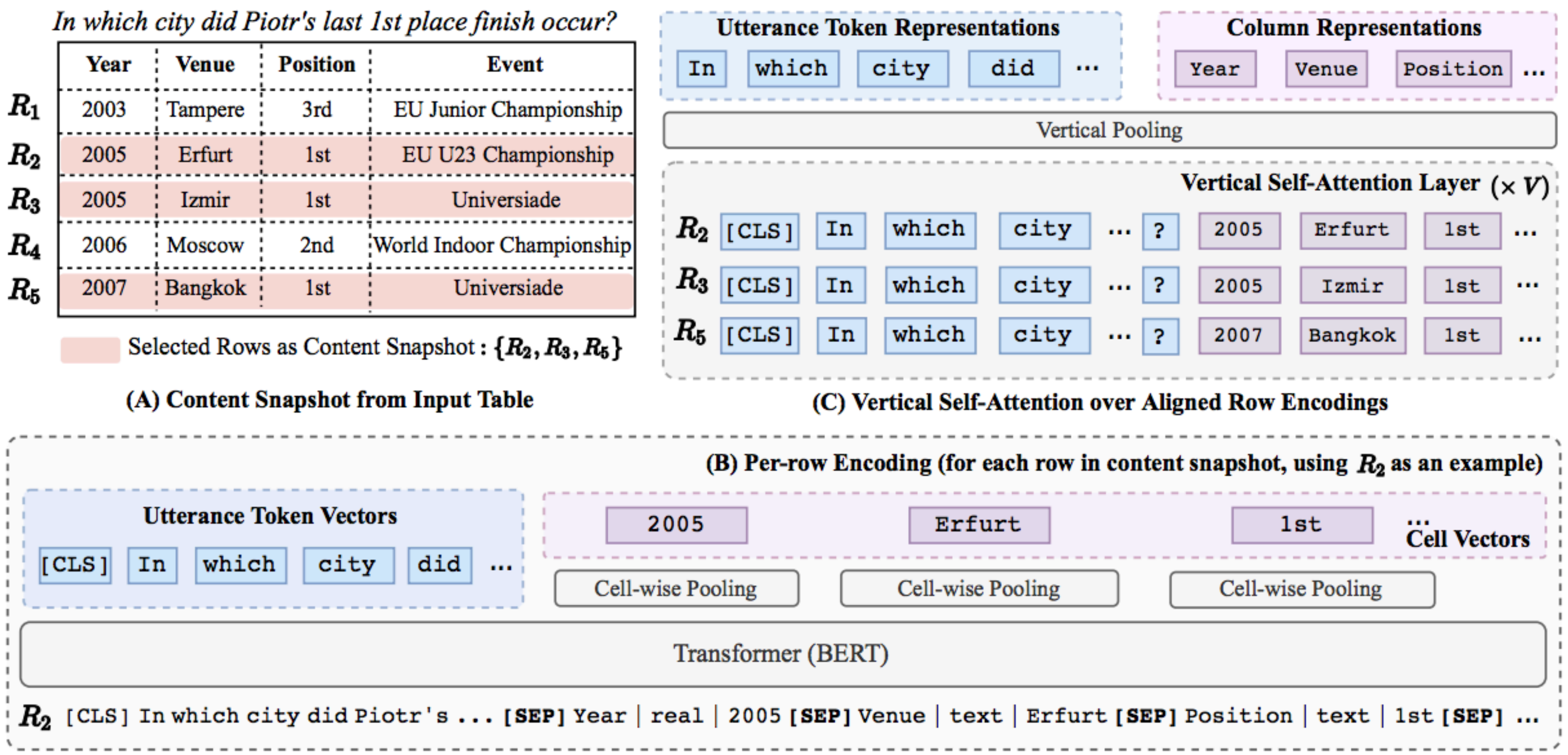
TAPAS: Weakly Supervised Table Parsing via Pre-training (Yin et al. 2020)

Challenges

- ***Objective:*** *standard MLM; Masked Column Prediction (MCP); Cell Value Recovery (CVR)*

- ***Objective:*** *standard MLM and relevant table prediction*

Learning objective does not explicitly enforce alignment between text and table

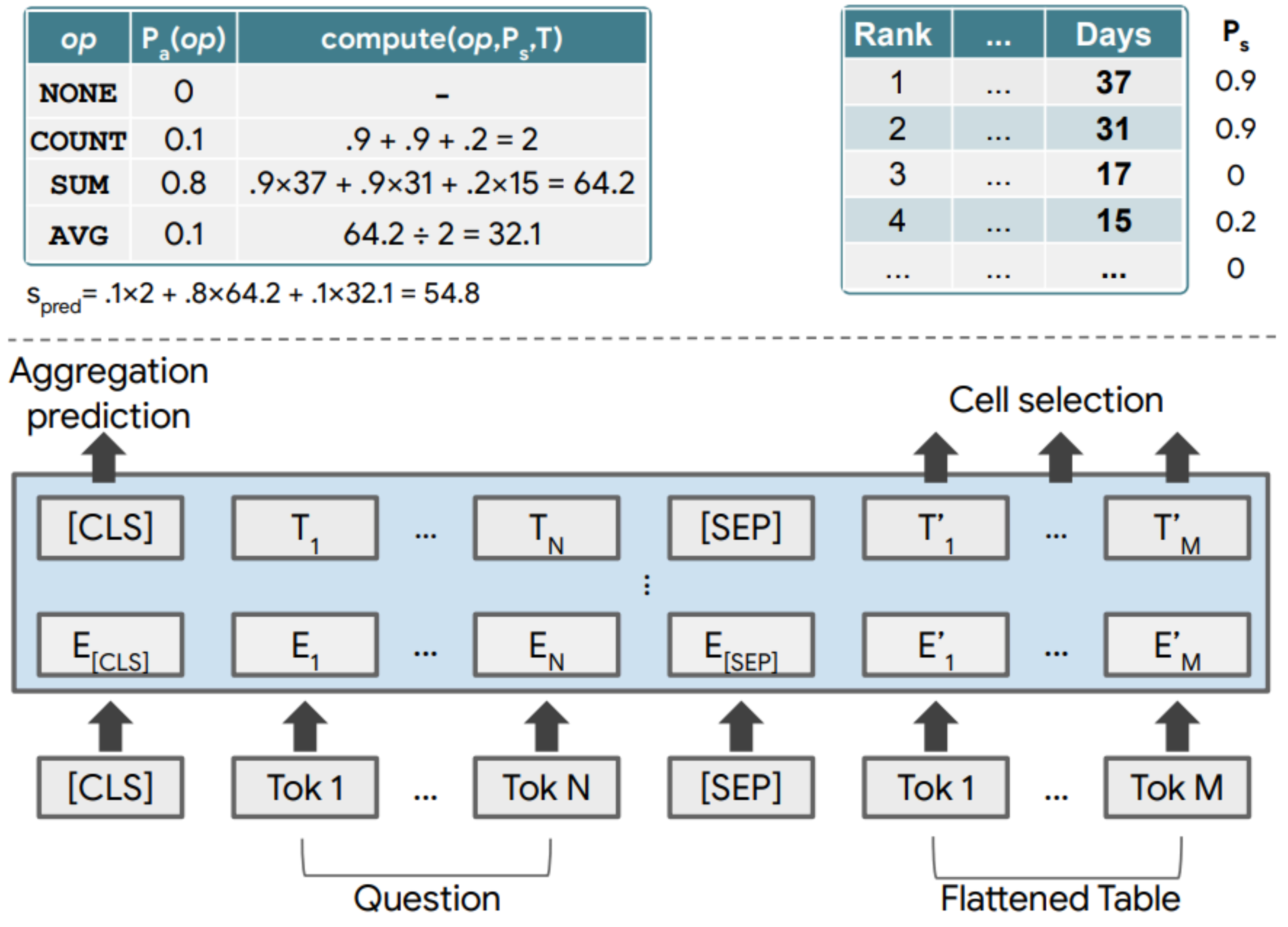


In which city did Piotr's last 1st place finish occur?

	Year	Venue	Position	Event
R_1	2003	Tampere	3rd	EU Junior Championship
R_2	2005	Erfurt	1st	EU U23 Championship
R_3	2005	Izmir	1st	Universiade
R_4	2006	Moscow	2nd	World Indoor Championship
R_5	2007	Bangkok	1st	Universiade

Selected Rows as Content Snapshot : $\{R_2, R_3, R_5\}$
(A) Content Snapshot from Input Table
Utterance Token Representations
In which city did ...
Column Representations
Year Venue Position ...
Vertical Pooling
Vertical Self-Attention Layer (× V)
R_2 [CLS] In which city ... ? 2005 Erfurt 1st ...
R_3 [CLS] In which city ... ? 2005 Izmir 1st ...
R_5 [CLS] In which city ... ? 2007 Bangkok 1st ...
(C) Vertical Self-Attention over Aligned Row Encodings
(B) Per-row Encoding (for each row in content snapshot, using R_2 as an example)
Utterance Token Vectors
[CLS] In which city did ...
2005 Erfurt 1st ...
Cell Vectors
Cell-wise Pooling
Cell-wise Pooling
Cell-wise Pooling
Transformer (BERT)
R_2 [CLS] In which city did Piotr's ... [SEP] Year | real | 2005 [SEP] Venue | text | Erfurt [SEP] Position | text | 1st [SEP] ...

TaBERT: Pretraining for Joint Understanding of Textual And Tabular Data (Yin et al. 2020)



op	$P_a(op)$	compute(op, P_s, T)
NONE	0	-
COUNT	0.1	.9 + .9 + .2 = 2
SUM	0.8	.9×37 + .9×31 + .2×15 = 64.2
AVG	0.1	64.2 ÷ 2 = 32.1

s_{pred} = .1×2 + .8×64.2 + .1×32.1 = 54.8

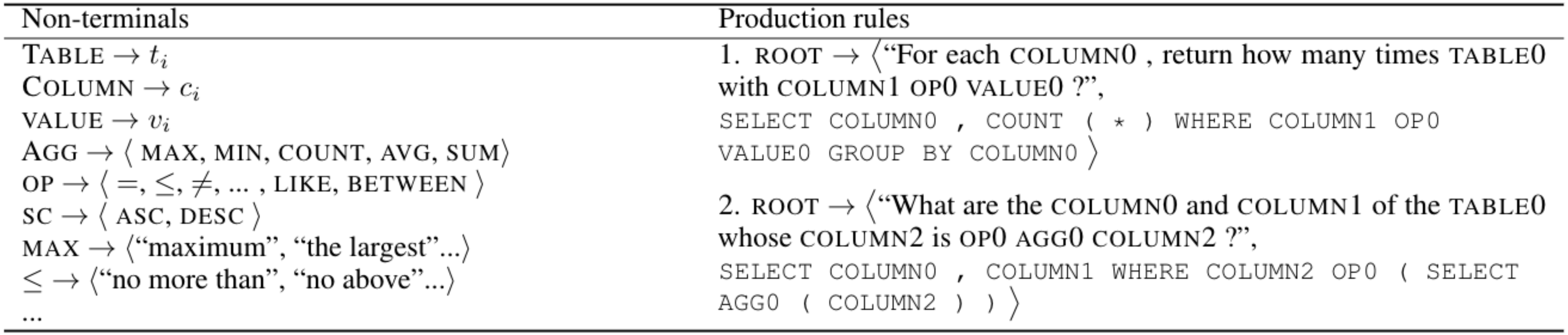
Rank	...	Days	P_s
1	...	37	0.9
2	...	31	0.9
3	...	17	0
4	...	15	0.2
...	...	...	0

Aggregation prediction
Cell selection
[CLS] T_1 ... T_N [SEP] T'_1 ... T'_M
$E_{[CLS]}$ E_1 ... E_N $E_{[SEP]}$ E'_1 ... E'_M
[CLS] Tok 1 ... Tok N [SEP] Tok 1 ... Tok M
Question
Flattened Table

TAPAS: Weakly Supervised Table Parsing via Pre-training (Yin et al. 2020)

Synthesize Text-to-SQL Data

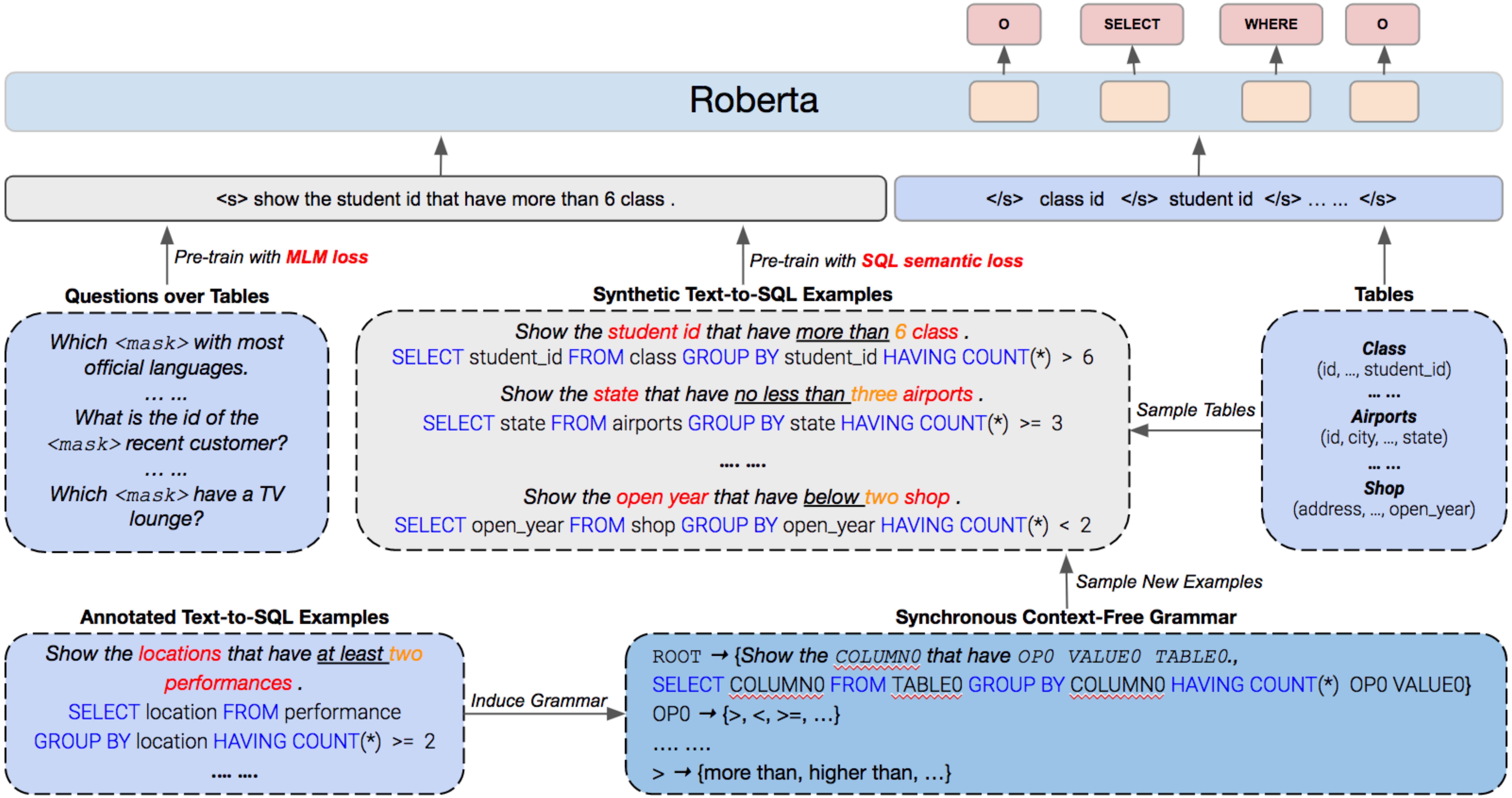
- Induce **synchronous context-free grammar (SCFG)** from existing text-to-SQL datasets.
- Synthesize text-SQL pairs from **high-quality tables (340k)** using the SCFG.
- Pre-train the model on the synthetic data using a **novel text-schema linking objective** that predicts the syntactic role of a table field in the SQL for each text-SQL pair
- Include **masked language objective (MLM) as a regularization** over existing **table-and-language modeling** datasets



Non-terminals	Production rules
TABLE → t_i COLUMN → c_i VALUE → v_i AGG → ⟨ MAX, MIN, COUNT, AVG, SUM⟩ OP → ⟨ =, ≤, ≠, ... , LIKE, BETWEEN ⟩ SC → ⟨ ASC, DESC ⟩ MAX → ⟨“maximum”, “the largest”...⟩ ≤ → ⟨“no more than”, “no above”...⟩ ...	1. ROOT → ⟨“For each COLUMN0 , return how many times TABLE0 with COLUMN1 OP0 VALUE0 ?”, `SELECT COLUMN0 , COUNT ( * ) WHERE COLUMN1 OP0 VALUE0 GROUP BY COLUMN0` ⟩ 2. ROOT → ⟨“What are the COLUMN0 and COLUMN1 of the TABLE0 whose COLUMN2 is OP0 AGG0 COLUMN2 ?”, `SELECT COLUMN0 , COLUMN1 WHERE COLUMN2 OP0 ( SELECT AGG0 ( COLUMN2 ) )` ⟩

GraPPa: Grammar-Augmented Pre-training for Table Semantic Parsing. Yu et al. 2020.

Grammar-Augmented Pre-training



O SELECT WHERE O

Roberta

<s> show the student id that have more than 6 class .

</s> class id </s> student id </s> … … </s>

Pre-train with **MLM loss**

Pre-train with **SQL semantic loss**

Questions over Tables

Which <mask> with most official languages.

… …

What is the id of the <mask> recent customer?

… …

Which <mask> have a TV lounge?

Synthetic Text-to-SQL Examples

Show the student id that have more than 6 class .

SELECT student_id FROM class GROUP BY student_id HAVING COUNT(*) > 6

Show the state that have no less than three airports .

SELECT state FROM airports GROUP BY state HAVING COUNT(*) >= 3

…. ….

Show the open year that have below two shop .

SELECT open_year FROM shop GROUP BY open_year HAVING COUNT(*) < 2

Tables

Class

(id, …, student_id)

… …

Airports

(id, city, …, state)

… …

Shop

(address, …, open_year)

Sample Tables

Sample New Examples

Annotated Text-to-SQL Examples

Show the locations that have at least two performances .

SELECT location FROM performance GROUP BY location HAVING COUNT(*) >= 2

…. ….

Induce Grammar

Synchronous Context-Free Grammar

ROOT → {*Show the COLUMN0 that have OP0 VALUE0 TABLE0.*,

SELECT COLUMN0 FROM TABLE0 GROUP BY COLUMN0 HAVING COUNT(*) OP0 VALUE0}

OP0 → {>, <, >=, …}

…. ….

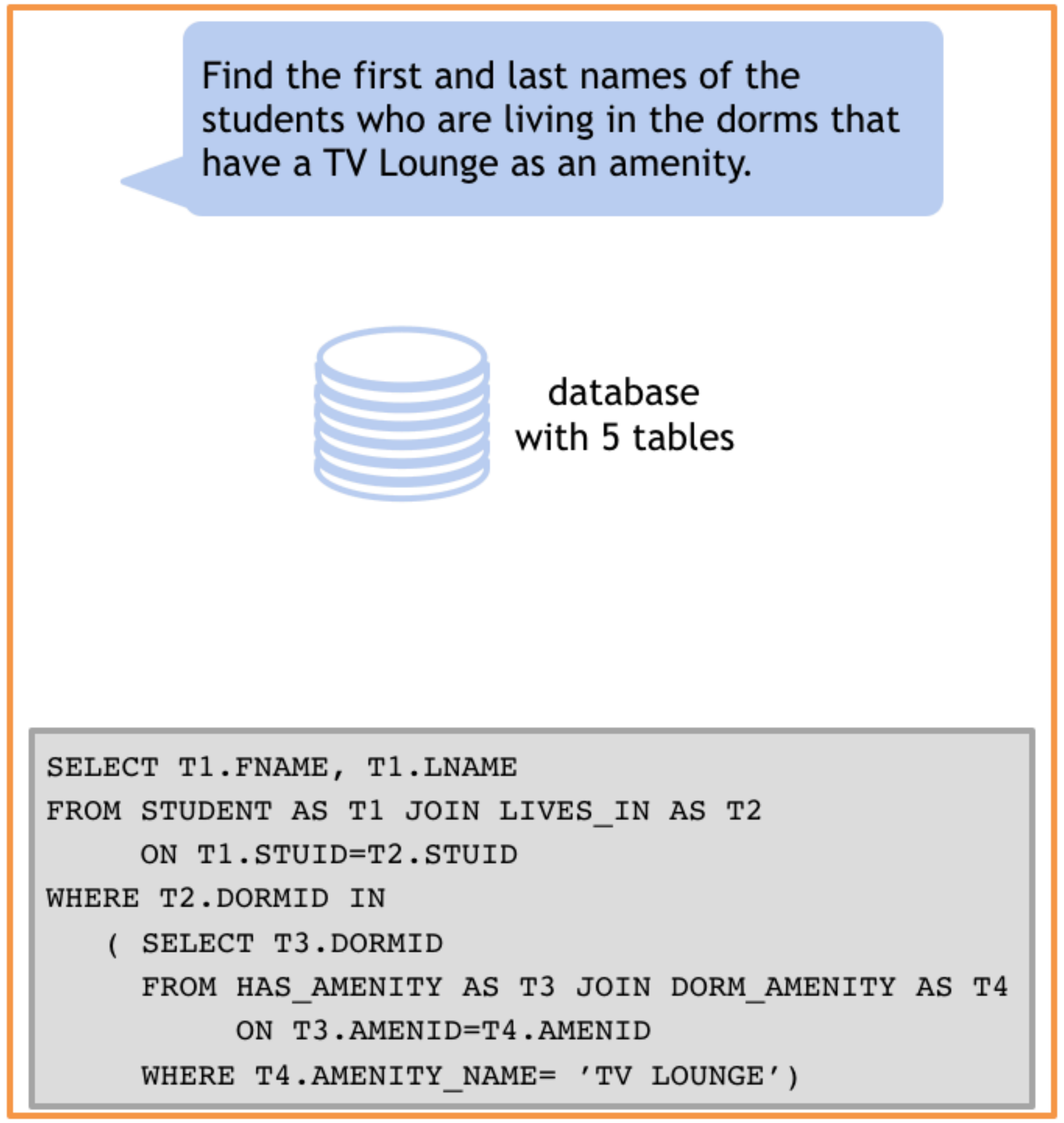
> → {more than, higher than, …}

GraPPa: Grammar-Augmented Pre-training for Table Semantic Parsing. Yu et al. 2020.

Experiments

Fully Supervised Semantic Parsing Tasks

Spider and **WikiSQL**

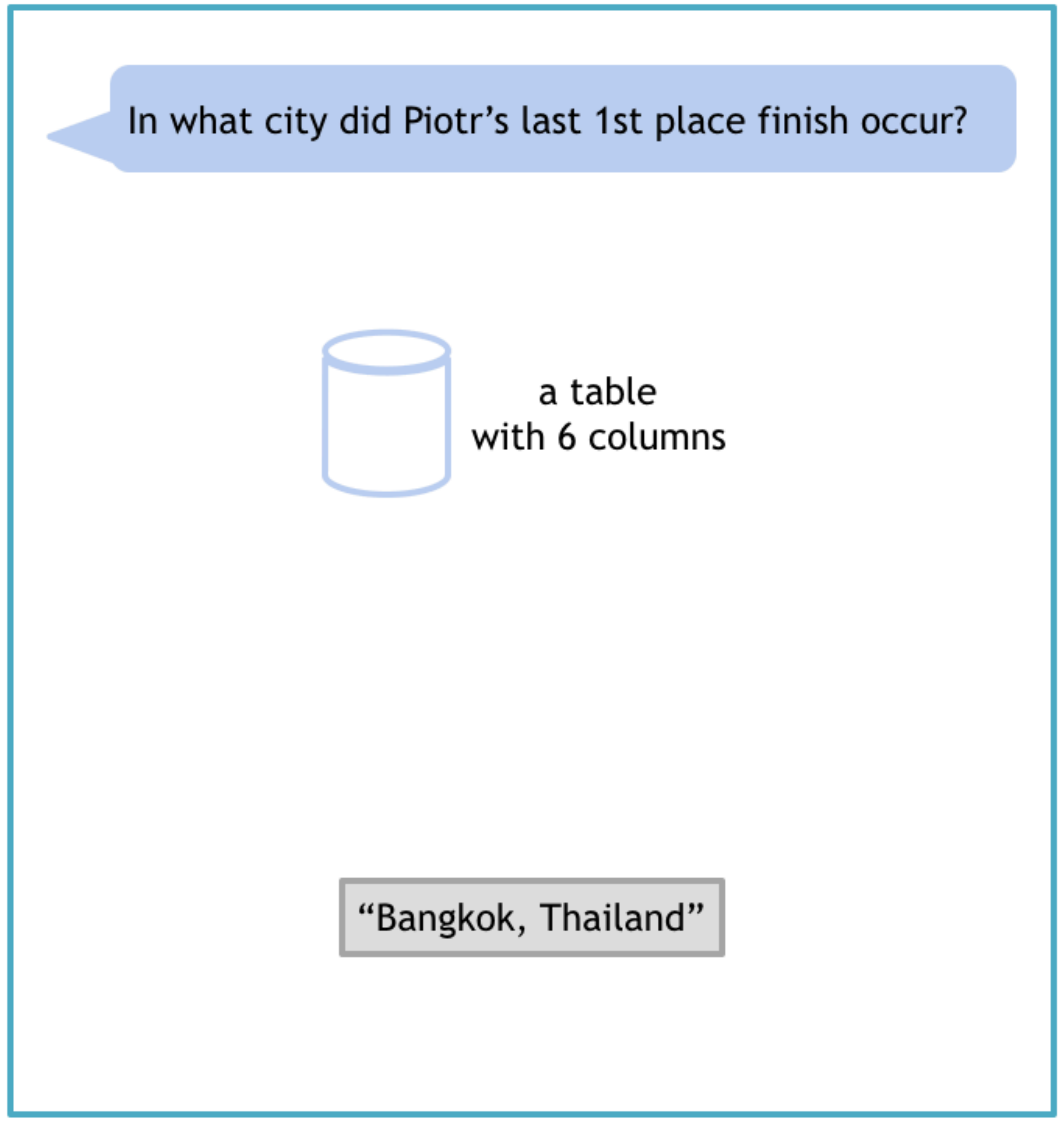


Find the first and last names of the students who are living in the dorms that have a TV Lounge as an amenity.

```
SELECT T1.FNAME, T1.LNAME
FROM STUDENT AS T1 JOIN LIVES_IN AS T2
     ON T1.STUID=T2.STUID
WHERE T2.DORMID IN
   ( SELECT T3.DORMID
     FROM HAS_AMENITY AS T3 JOIN DORM_AMENITY AS T4
          ON T3.AMENID=T4.AMENID
     WHERE T4.AMENITY_NAME= ’TV LOUNGE’)
```

Weakly Supervised Semantic Parsing Tasks

WikiTQ and **WikiSQL**



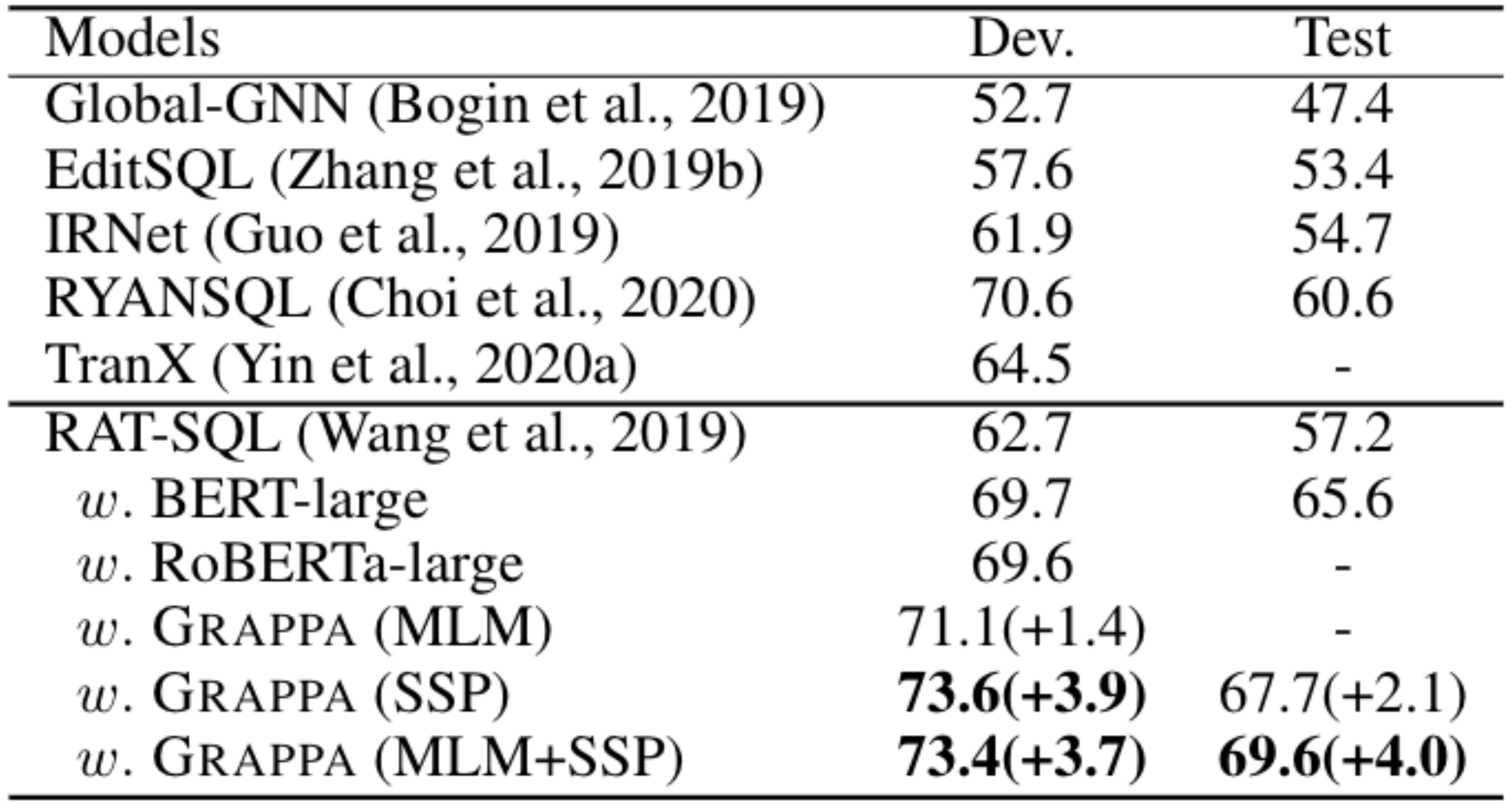
In what city did Piotr’s last 1st place finish occur?

“Bangkok, Thailand”

WikiTableQuestions (Pasupat and Liang. 2015); WikiSQL (Zhong et al. 2017); Spider (Yu et al. 2018)

Fully Supervised Semantic Parsing Results

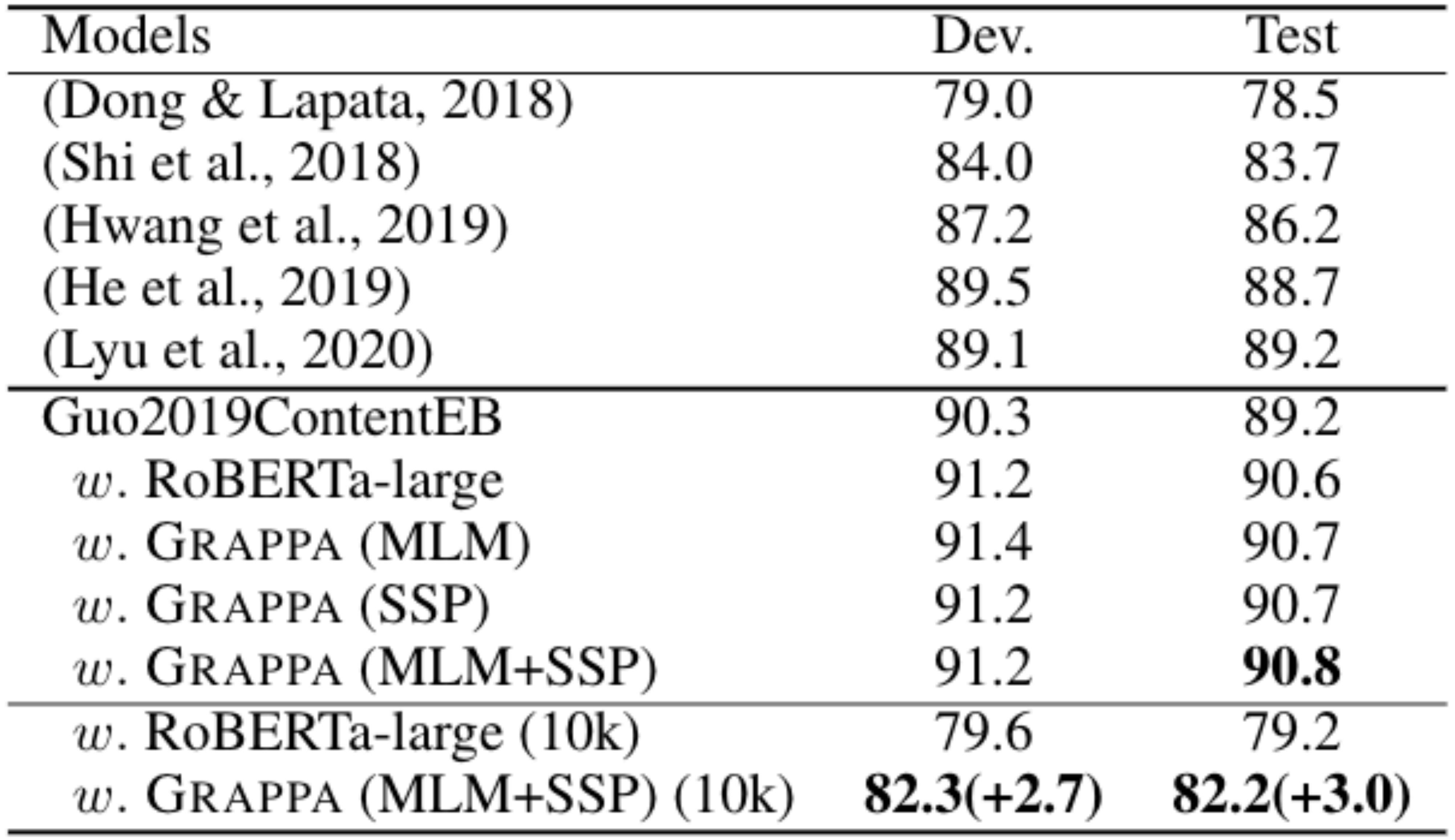
Our best model GraPPa (MLN+SSP) **achieves new state-of-the-art performance**, surpassing previous work by a **margin of 4%**



Models	Dev.	Test
Global-GNN (Bogin et al., 2019)	52.7	47.4
EditSQL (Zhang et al., 2019b)	57.6	53.4
IRNet (Guo et al., 2019)	61.9	54.7
RYANSQL (Choi et al., 2020)	70.6	60.6
TranX (Yin et al., 2020a)	64.5	-
RAT-SQL (Wang et al., 2019)	62.7	57.2
w. BERT-large	69.7	65.6
w. RoBERTa-large	69.6	-
w. GRAPPA (MLM)	71.1(+1.4)	-
w. GRAPPA (SSP)	**73.6(+3.9)**	67.7(+2.1)
w. GRAPPA (MLM+SSP)	**73.4(+3.7)**	**69.6(+4.0)**

Spider Results

Our best model GraPPa (MLN+SSP) **achieves new state-of-the-art performance.** The improvement from the base model is even more significant when there is **less training data**.



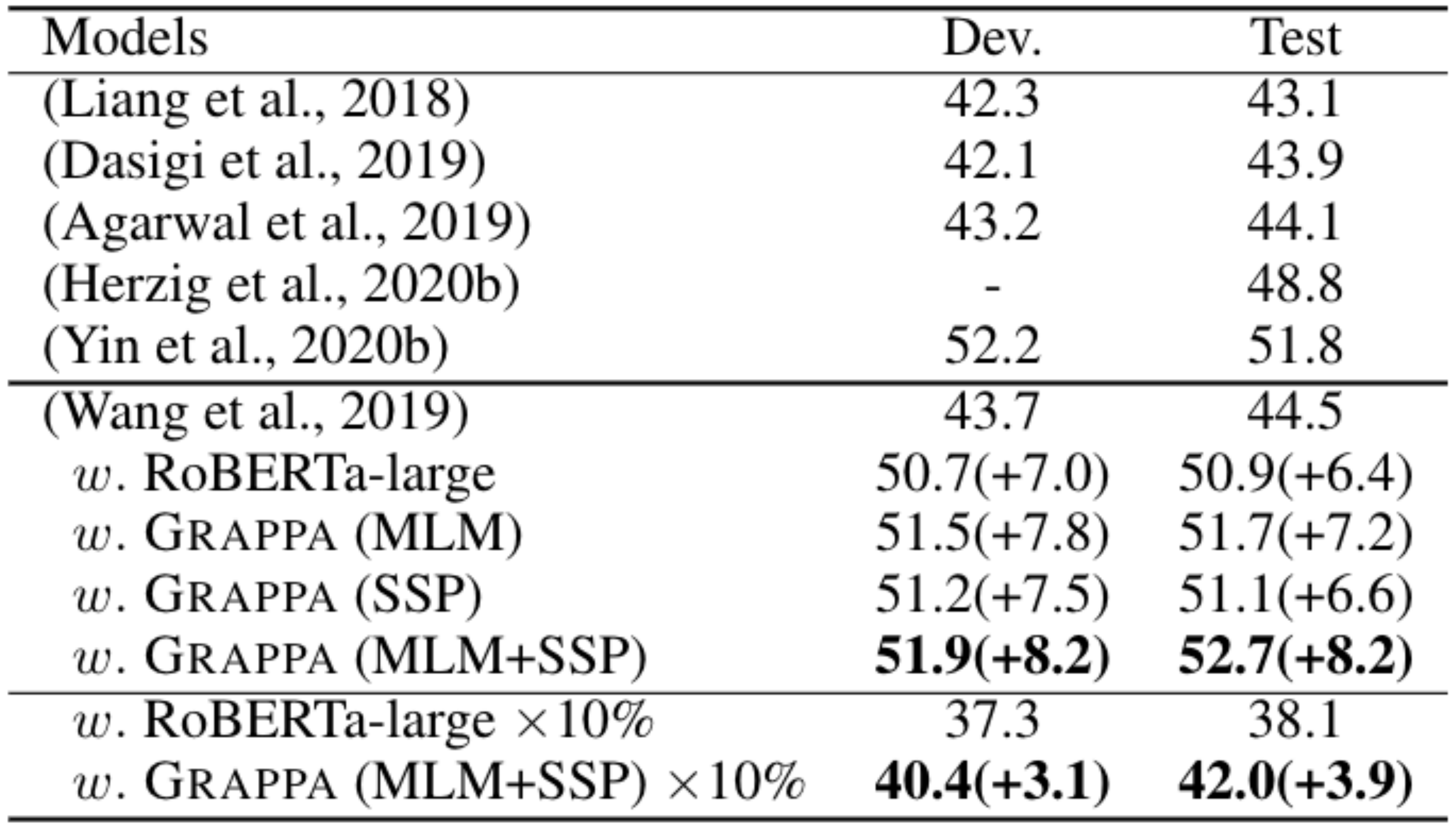
Models	Dev.	Test
(Dong & Lapata, 2018)	79.0	78.5
(Shi et al., 2018)	84.0	83.7
(Hwang et al., 2019)	87.2	86.2
(He et al., 2019)	89.5	88.7
(Lyu et al., 2020)	89.1	89.2
Guo2019ContentEB	90.3	89.2
w. RoBERTa-large	91.2	90.6
w. GRAPPA (MLM)	91.4	90.7
w. GRAPPA (SSP)	91.2	90.7
w. GRAPPA (MLM+SSP)	91.2	**90.8**
w. RoBERTa-large (10k)	79.6	79.2
w. GRAPPA (MLM+SSP) (10k)	**82.3(+2.7)**	**82.2(+3.0)**

WikiSQL Results

GraPPa: Grammar-Augmented Pre-training for Table Semantic Parsing. Yu et al. 2020.

Weakly Supervised Semantic Parsing Results

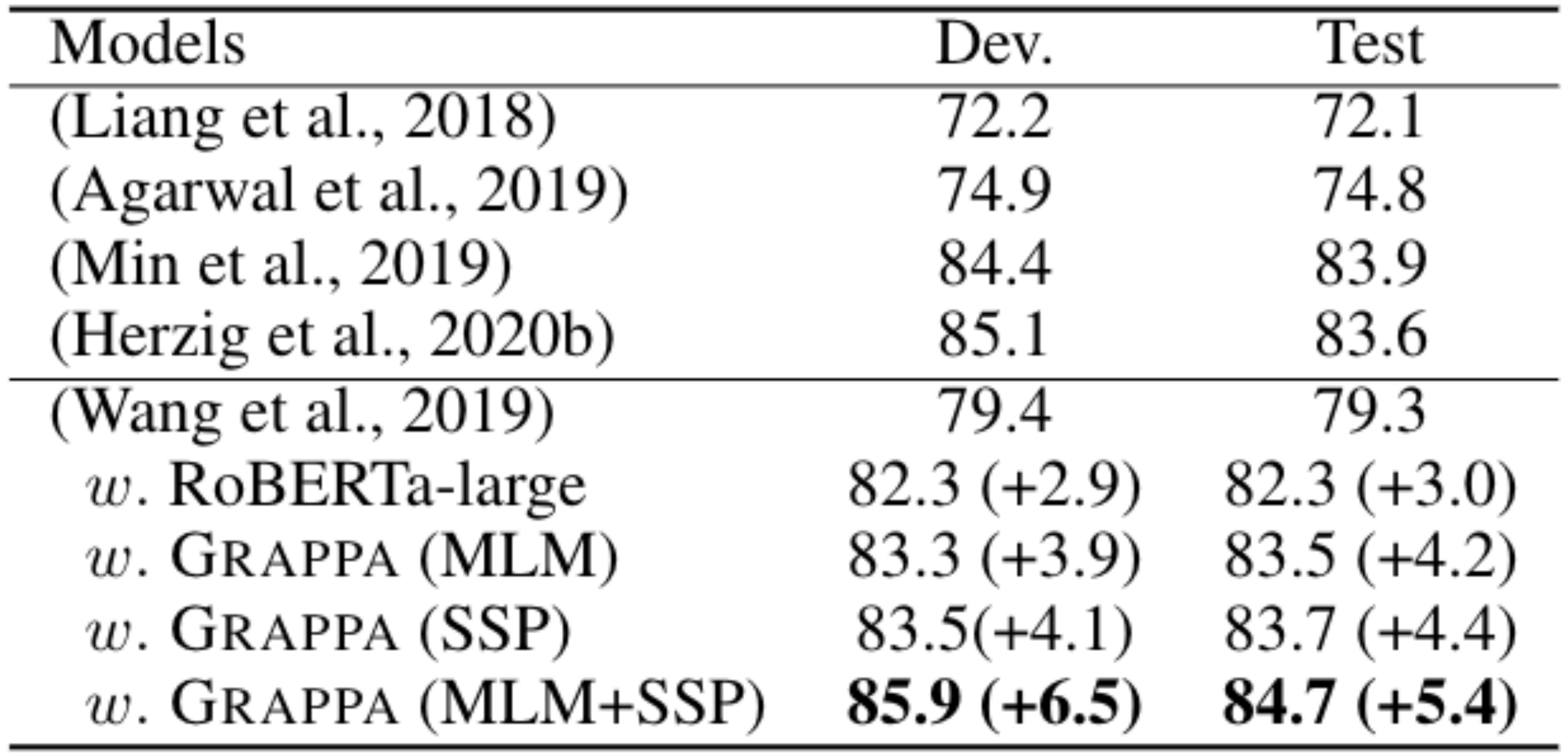
Our best model GraPPa (MLN+SSP) **achieves new state-of-the-art performance**, improve from RoBERTa by a **margin of 1.8%.** The improvement is even more significant using 10% of the training data.



Models	Dev.	Test
(Liang et al., 2018)	42.3	43.1
(Dasigi et al., 2019)	42.1	43.9
(Agarwal et al., 2019)	43.2	44.1
(Herzig et al., 2020b)	-	48.8
(Yin et al., 2020b)	52.2	51.8
(Wang et al., 2019)	43.7	44.5
w. RoBERTa-large	50.7(+7.0)	50.9(+6.4)
w. GRAPPA (MLM)	51.5(+7.8)	51.7(+7.2)
w. GRAPPA (SSP)	51.2(+7.5)	51.1(+6.6)
w. GRAPPA (MLM+SSP)	**51.9(+8.2)**	**52.7(+8.2)**
w. RoBERTa-large ×10%	37.3	38.1
w. GRAPPA (MLM+SSP) ×10%	**40.4(+3.1)**	**42.0(+3.9)**

WikiTableQuestions Results

Our best model GraPPa (MLN+SSP) **achieves new state-of-the-art performance.**

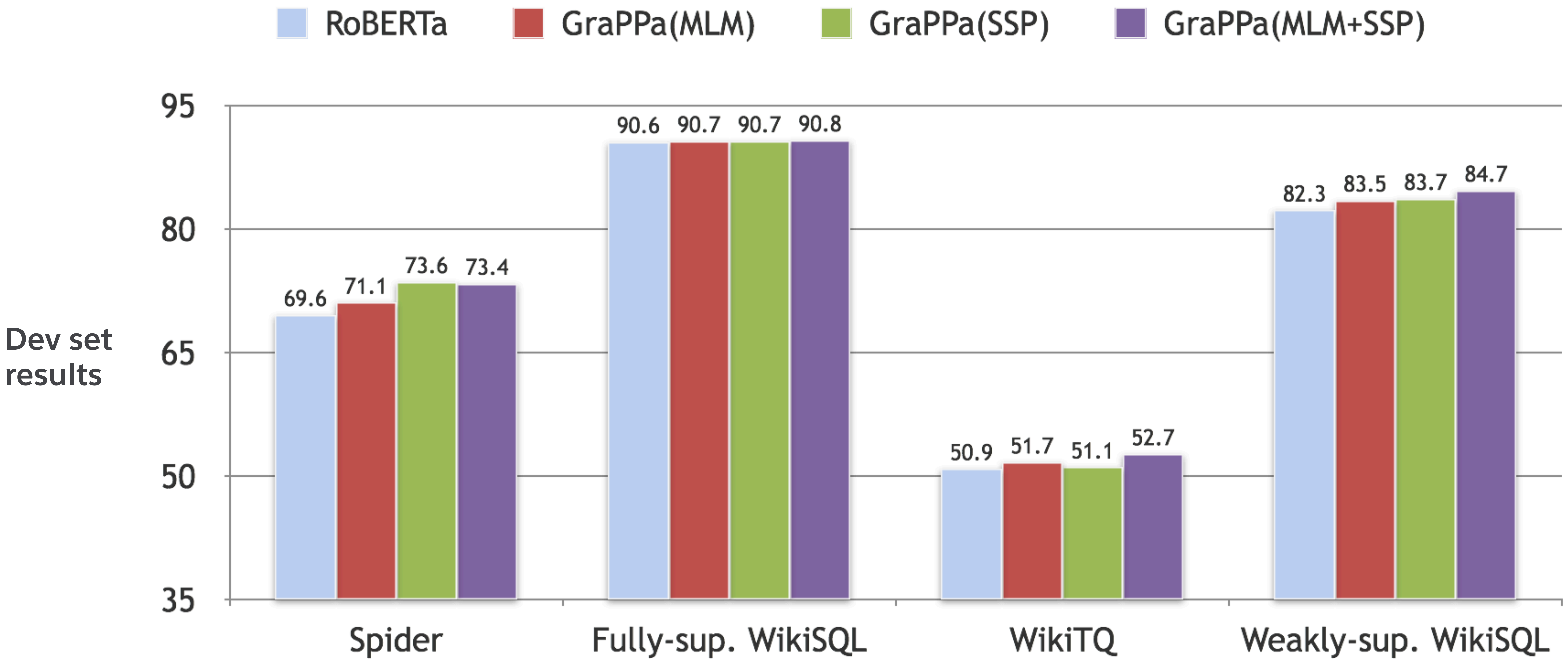


Models	Dev.	Test
(Liang et al., 2018)	72.2	72.1
(Agarwal et al., 2019)	74.9	74.8
(Min et al., 2019)	84.4	83.9
(Herzig et al., 2020b)	85.1	83.6
(Wang et al., 2019)	79.4	79.3
w. RoBERTa-large	82.3 (+2.9)	82.3 (+3.0)
w. GRAPPA (MLM)	83.3 (+3.9)	83.5 (+4.2)
w. GRAPPA (SSP)	83.5(+4.1)	83.7 (+4.4)
w. GRAPPA (MLM+SSP)	**85.9 (+6.5)**	**84.7 (+5.4)**

Weakly Supervised WikiSQL Results

GraPPa: Grammar-Augmented Pre-training for Table Semantic Parsing. Yu et al. 2020.

Effect of Different Pre-training Objectives



GraPPa: Grammar-Augmented Pre-training for Table Semantic Parsing. Yu et al. 2020.

Takeaway

- GraPPa is an effective pre-training approach for table semantic parsing.

Takeaway

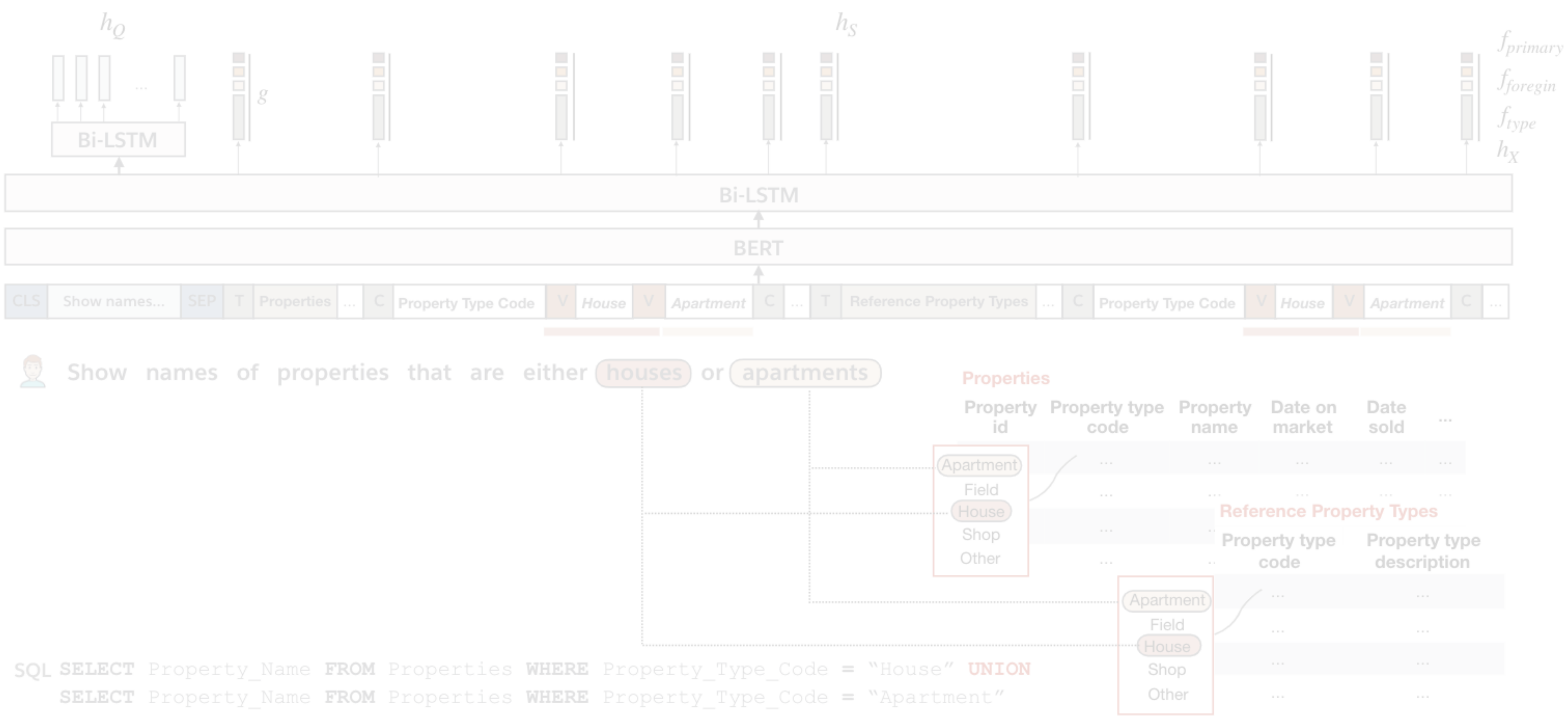
- GraPPa is an effective pre-training approach for table semantic parsing.
- It learns a compositional inductive bias in the joint representations of textual and tabular data via a novel text-schema linking objective over synthesized question-SQL pairs.

Takeaway

- GraPPa is an effective pre-training approach for table semantic parsing.
- It learns a compositional inductive bias in the joint representations of textual and tabular data via a novel text-schema linking objective over synthesized question-SQL pairs.
- On four popular fully supervised and weakly supervised table semantic parsing benchmarks, GRAPPA significantly outperforms RoBERTa-LARGE as the feature representation layers and establishes new state-of-the- art results on all of them.

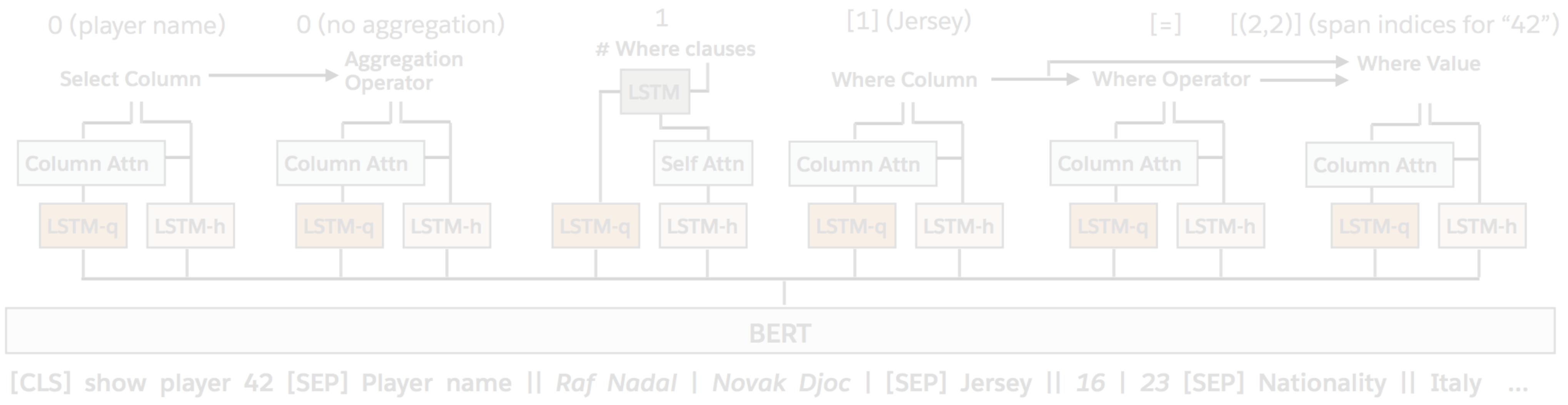
I. Content-Aware Textual-Tabular Encodings for Table Semantic Parsing (TSP)

Bridging Textual and Tabular Data for Cross-Domain Text-to-SQL Semantic Parsing. Lin et al. 2020.



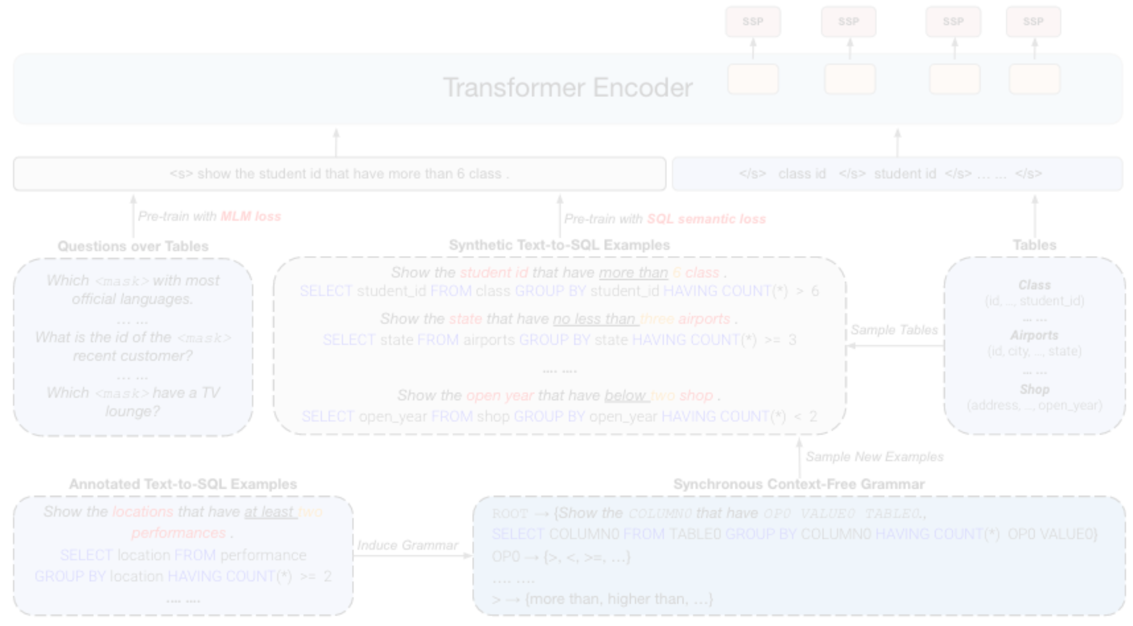
```
SQL SELECT Property_Name FROM Properties WHERE Property_Type_Code = "House" UNION
    SELECT Property_Name FROM Properties WHERE Property_Type_Code = "Apartment"
```

ColloQL: Robust Cross-Domain Text-to-SQL over Search Queries. Radhakrishnan et al. 2020.



II. Pre-training Textual-Tabular Representations with Semantic Scaffolds

GraPPa: Grammar-Augmented Pre-training for Table Semantic Parsing. Yu et al. 2020.

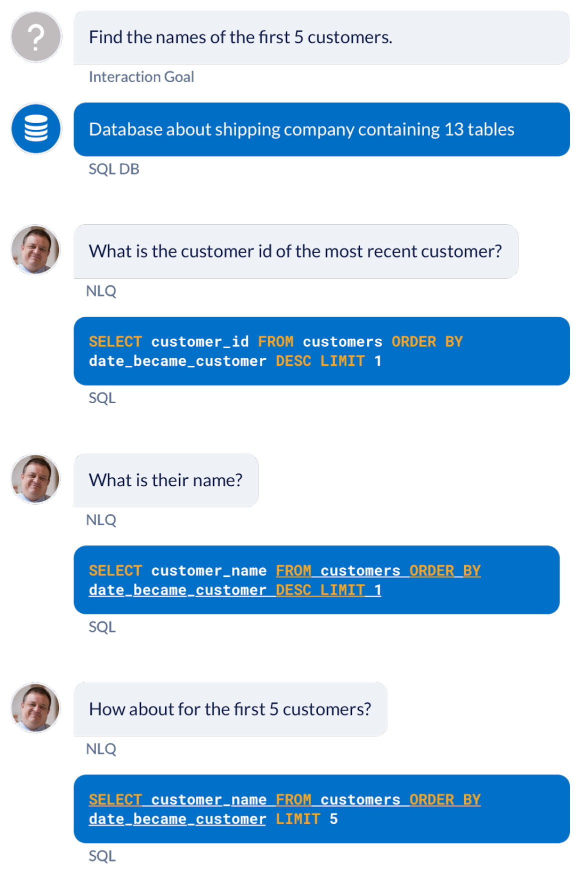

SSP
SSP
SSP
SSP
Transformer Encoder
<s> show the student id that have more than 6 class .
</s> class id </s> student id </s> ... ... </s>
Pre-train with MLM loss
Pre-train with SQL semantic loss
Questions over Tables
Which <mask> with most official languages.
What is the id of the <mask> recent customer?
Which <mask> have a TV lounge?
Synthetic Text-to-SQL Examples
Show the student id that have more than 6 class .
SELECT student_id FROM class GROUP BY student_id HAVING COUNT(*) > 6
Show the state that have no less than three airports .
SELECT state FROM airports GROUP BY state HAVING COUNT(*) >= 3
Show the open year that have below two shop .
SELECT open_year FROM shop GROUP BY open_year HAVING COUNT(*) < 2
Sample Tables
Tables
Class
(id, ..., student_id)
Airports
(id, city, ..., state)
Shop
(address, ..., open_year)
Sample New Examples
Annotated Text-to-SQL Examples
Show the locations that have at least two performances .
SELECT location FROM performance GROUP BY location HAVING COUNT(*) >= 2
Induce Grammar
Synchronous Context-Free Grammar
ROOT → {Show the COLUMN0 that have OP0 VALUE0 TABLE0.,
SELECT COLUMN0 FROM TABLE0 GROUP BY COLUMN0 HAVING COUNT(*) OP0 VALUE0}
OP0 → {>, <, >=, ...}
> → {more than, higher than, ...}

III. Conversational Table Semantic Parsing

SParC: Cross-Domain Semantic Parsing in Context. Yu et al. 2019.

Editing-Based SQL Query Generation for Cross-Domain Context-Dependent Questions. Zhang et al. 2019.

CoSQL: A Conversational Text-to-SQL Challenge Towards Cross-Domain Natural Language Interfaces to Databases. Yu et al. 2019.



Find the names of the first 5 customers.

Interaction Goal

Database about shipping company containing 13 tables

SQL DB

What is the customer id of the most recent customer?

NLQ

```
SELECT customer_id FROM customers ORDER BY
date_became_customer DESC LIMIT 1
```

SQL

What is their name?

NLQ

```
SELECT customer_name FROM customers ORDER BY
date_became_customer DESC LIMIT 1
```

SQL

How about for the first 5 customers?

NLQ

```
SELECT customer_name FROM customers ORDER BY
date_became_customer LIMIT 5
```

SQL

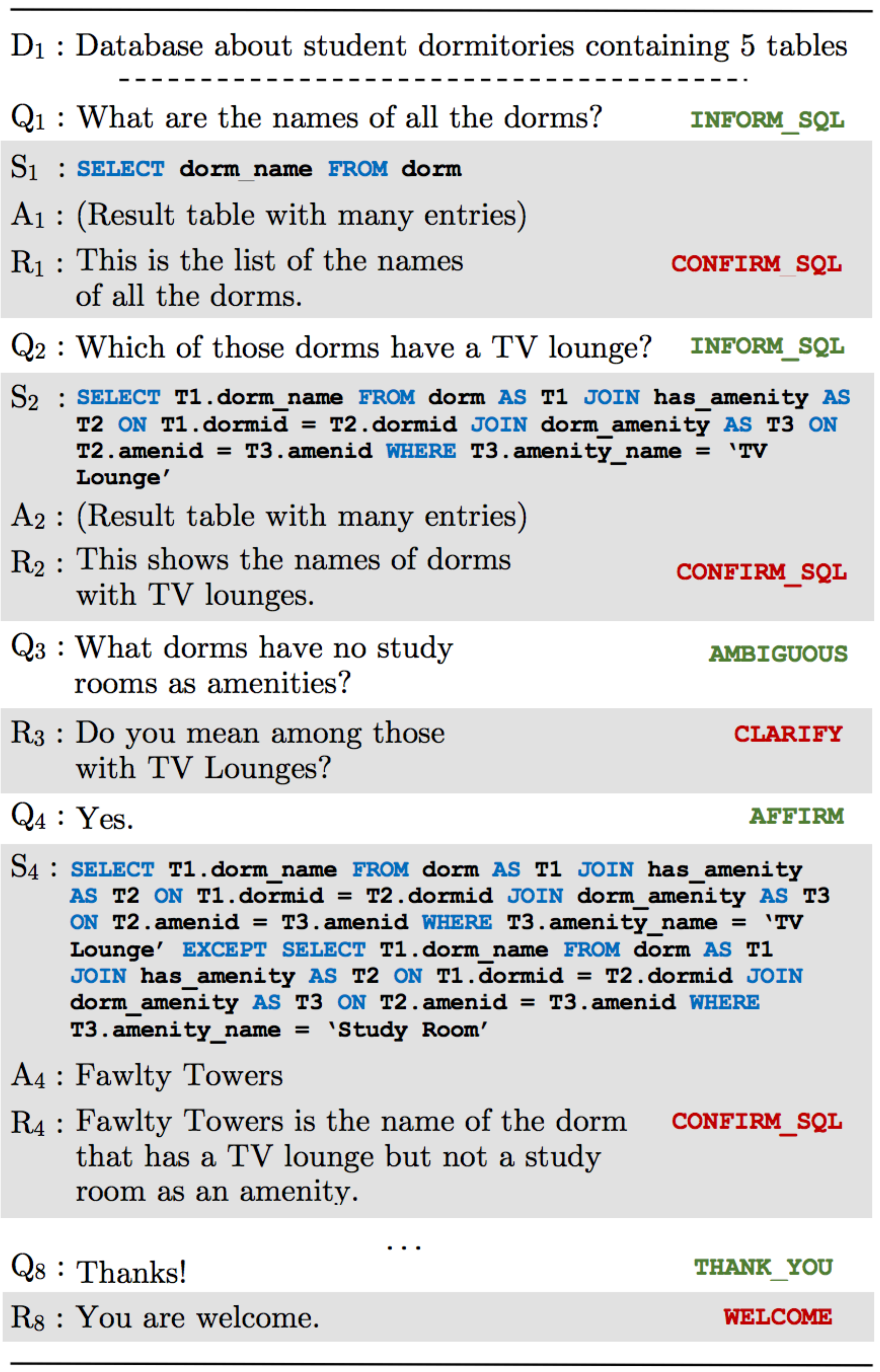
CoSQL: A Conversational Text-to-SQL Challenge

In real world, users typically issue sequences of questions when querying a database

Context-dependent utterances reflect special linguistic phenomena such as co-references and omission

Besides well-formed **information seeking** questions, users may issue utterances that require **clarification** and trigger **other dialogue actions**

System responses are better to be paired w/ accessible **natural language responses**



D_1 : Database about student dormitories containing 5 tables

Q_1 : What are the names of all the dorms? INFORM_SQL

S_1 : `SELECT dorm_name FROM dorm`

A_1 : (Result table with many entries)

R_1 : This is the list of the names of all the dorms. CONFIRM_SQL

Q_2 : Which of those dorms have a TV lounge? INFORM_SQL

S_2 :
```
SELECT T1.dorm_name FROM dorm AS T1 JOIN has_amenity AS
T2 ON T1.dormid = T2.dormid JOIN dorm_amenity AS T3 ON
T2.amenid = T3.amenid WHERE T3.amenity_name = 'TV
Lounge'
```

A_2 : (Result table with many entries)

R_2 : This shows the names of dorms with TV lounges. CONFIRM_SQL

Q_3 : What dorms have no study rooms as amenities? AMBIGUOUS

R_3 : Do you mean among those with TV Lounges? CLARIFY

Q_4 : Yes. AFFIRM

S_4 :
```
SELECT T1.dorm_name FROM dorm AS T1 JOIN has_amenity
AS T2 ON T1.dormid = T2.dormid JOIN dorm_amenity AS T3
ON T2.amenid = T3.amenid WHERE T3.amenity_name = 'TV
Lounge' EXCEPT SELECT T1.dorm_name FROM dorm AS T1
JOIN has_amenity AS T2 ON T1.dormid = T2.dormid JOIN
dorm_amenity AS T3 ON T2.amenid = T3.amenid WHERE
T3.amenity_name = 'Study Room'
```

A_4 : Fawlty Towers

R_4 : Fawlty Towers is the name of the dorm that has a TV lounge but not a study room as an amenity. CONFIRM_SQL

. . .

Q_8 : Thanks! THANK_YOU

R_8 : You are welcome. WELCOME

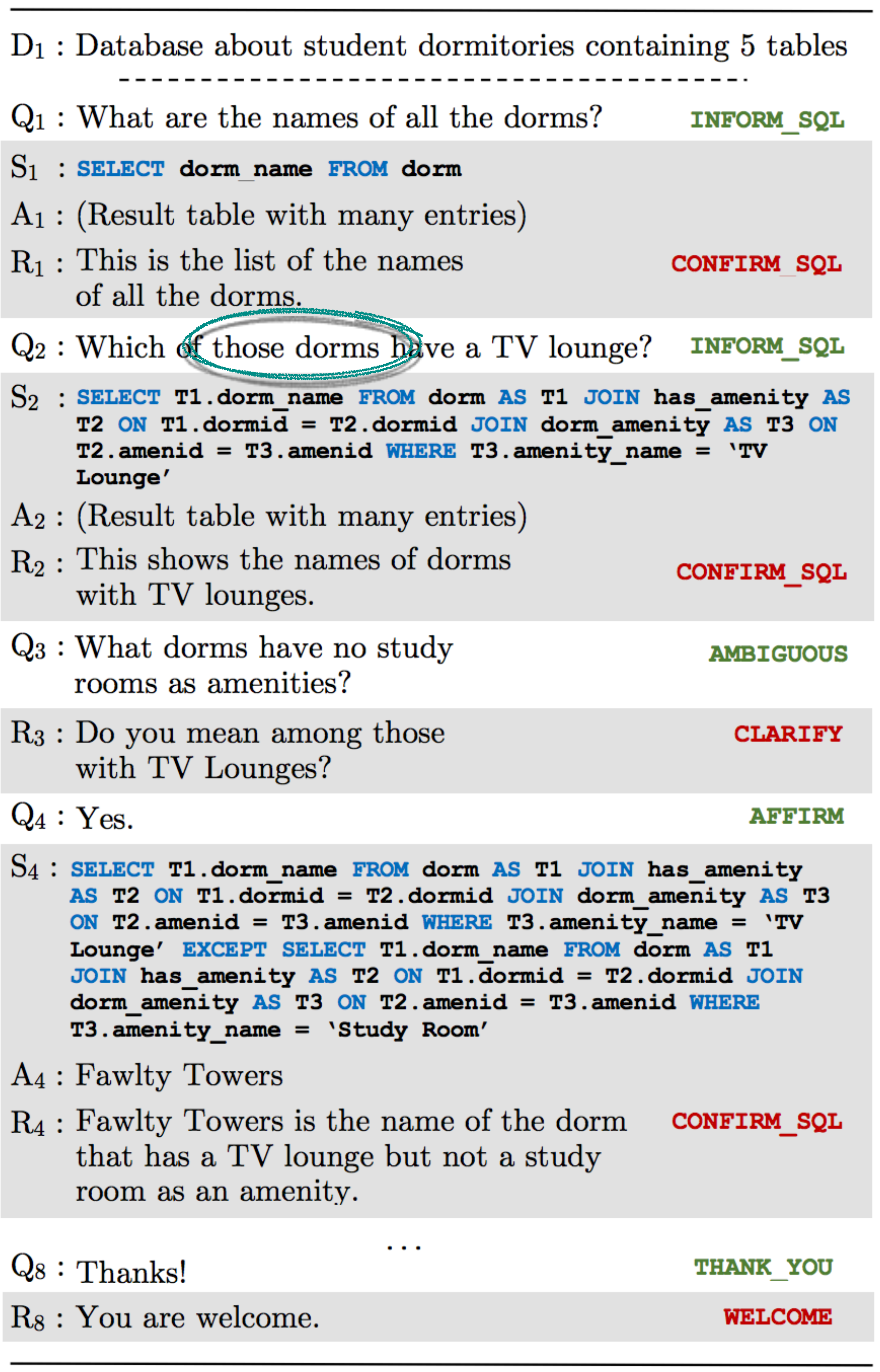
CoSQL: A Conversational Text-to-SQL Challenge

In real world, users typically issue sequences of questions when querying a database

Context-dependent utterances reflect special linguistic phenomena such as co-references and omission

Besides well-formed **information seeking** questions, users may issue utterances that require **clarification** and trigger **other dialogue actions**

System responses are better to be paired w/ accessible **natural language responses**



D_1 : Database about student dormitories containing 5 tables

Q_1 : What are the names of all the dorms? INFORM_SQL

S_1 : `SELECT dorm_name FROM dorm`

A_1 : (Result table with many entries)

R_1 : This is the list of the names of all the dorms. CONFIRM_SQL

Q_2 : Which of those dorms have a TV lounge? INFORM_SQL

S_2 : `SELECT T1.dorm_name FROM dorm AS T1 JOIN has_amenity AS T2 ON T1.dormid = T2.dormid JOIN dorm_amenity AS T3 ON T2.amenid = T3.amenid WHERE T3.amenity_name = 'TV Lounge'`

A_2 : (Result table with many entries)

R_2 : This shows the names of dorms with TV lounges. CONFIRM_SQL

Q_3 : What dorms have no study rooms as amenities? AMBIGUOUS

R_3 : Do you mean among those with TV Lounges? CLARIFY

Q_4 : Yes. AFFIRM

S_4 : `SELECT T1.dorm_name FROM dorm AS T1 JOIN has_amenity AS T2 ON T1.dormid = T2.dormid JOIN dorm_amenity AS T3 ON T2.amenid = T3.amenid WHERE T3.amenity_name = 'TV Lounge' EXCEPT SELECT T1.dorm_name FROM dorm AS T1 JOIN has_amenity AS T2 ON T1.dormid = T2.dormid JOIN dorm_amenity AS T3 ON T2.amenid = T3.amenid WHERE T3.amenity_name = 'Study Room'`

A_4 : Fawlty Towers

R_4 : Fawlty Towers is the name of the dorm that has a TV lounge but not a study room as an amenity. CONFIRM_SQL

. . .

Q_8 : Thanks! THANK_YOU

R_8 : You are welcome. WELCOME

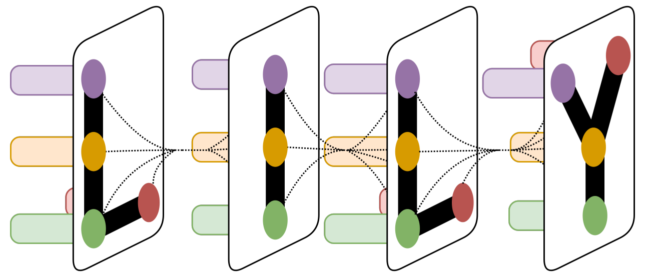
CoSQL: A Conversational Text-to-SQL Challenge

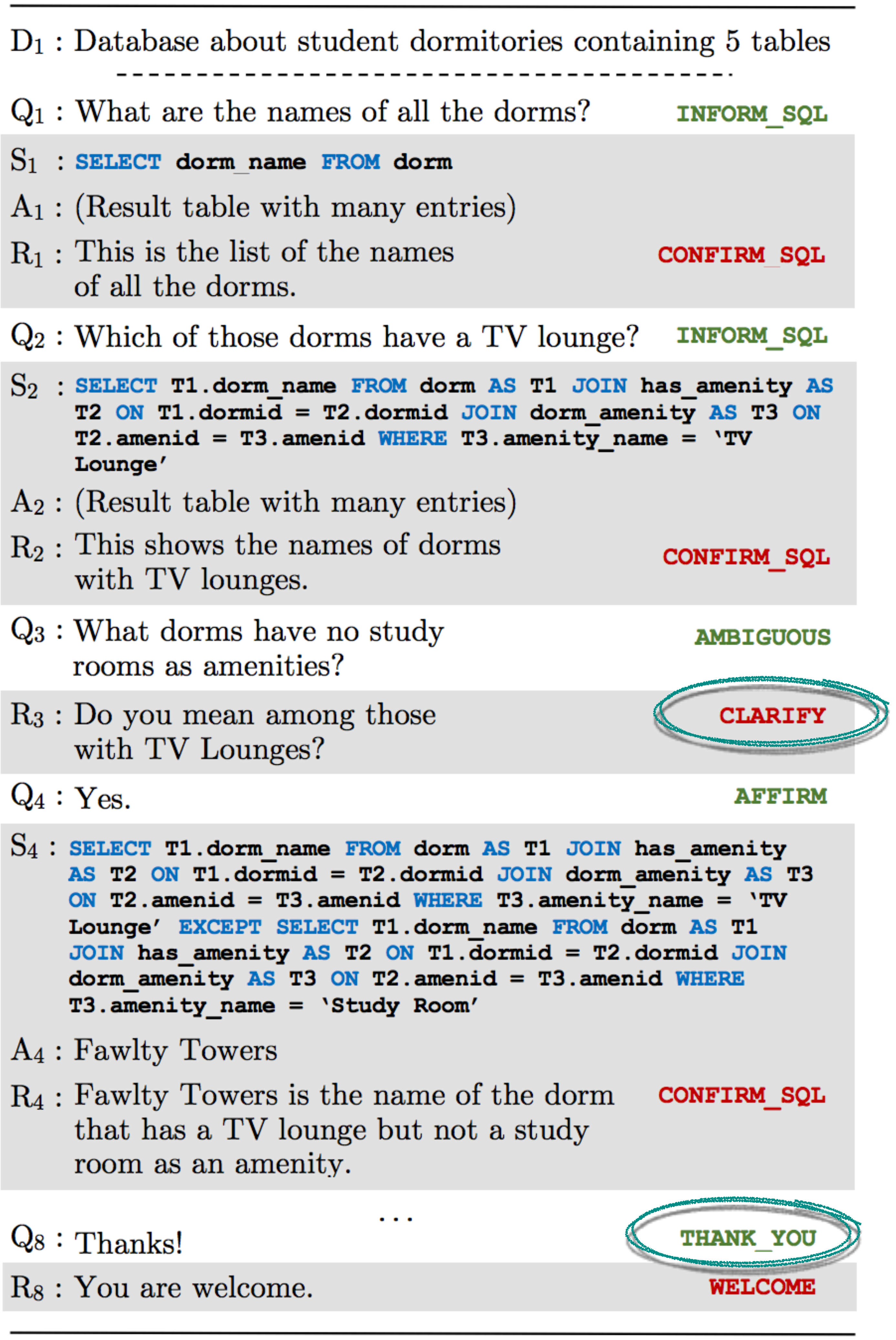
In real world, users typically issue sequences of questions when querying a database

Context-dependent utterances reflect special linguistic phenomena such as co-references and omission

Besides well-formed **information seeking** questions, users may issue utterances that require **clarification** and trigger **other dialogue actions**

System responses are better to be paired w/ accessible **natural language responses**





D_1 : Database about student dormitories containing 5 tables

Q_1 : What are the names of all the dorms? INFORM_SQL

S_1 :
```
SELECT dorm_name FROM dorm
```

A_1 : (Result table with many entries)

R_1 : This is the list of the names of all the dorms. CONFIRM_SQL

Q_2 : Which of those dorms have a TV lounge? INFORM_SQL

S_2 :
```
SELECT T1.dorm_name FROM dorm AS T1 JOIN has_amenity AS
T2 ON T1.dormid = T2.dormid JOIN dorm_amenity AS T3 ON
T2.amenid = T3.amenid WHERE T3.amenity_name = 'TV
Lounge'
```

A_2 : (Result table with many entries)

R_2 : This shows the names of dorms with TV lounges. CONFIRM_SQL

Q_3 : What dorms have no study rooms as amenities? AMBIGUOUS

R_3 : Do you mean among those with TV Lounges? CLARIFY

Q_4 : Yes. AFFIRM

S_4 :
```
SELECT T1.dorm_name FROM dorm AS T1 JOIN has_amenity
AS T2 ON T1.dormid = T2.dormid JOIN dorm_amenity AS T3
ON T2.amenid = T3.amenid WHERE T3.amenity_name = 'TV
Lounge' EXCEPT SELECT T1.dorm_name FROM dorm AS T1
JOIN has_amenity AS T2 ON T1.dormid = T2.dormid JOIN
dorm_amenity AS T3 ON T2.amenid = T3.amenid WHERE
T3.amenity_name = 'Study Room'
```

A_4 : Fawlty Towers

R_4 : Fawlty Towers is the name of the dorm that has a TV lounge but not a study room as an amenity. CONFIRM_SQL

. . .

Q_8 : Thanks! THANK_YOU

R_8 : You are welcome. WELCOME

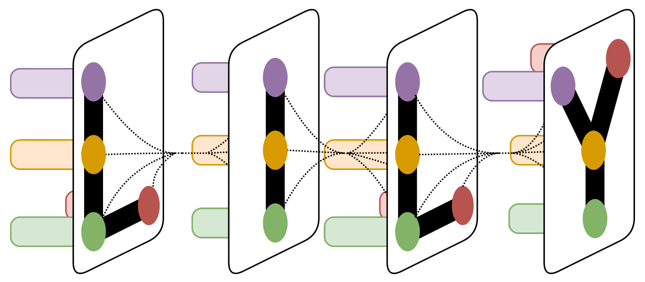
CoSQL: A Conversational Text-to-SQL Challenge

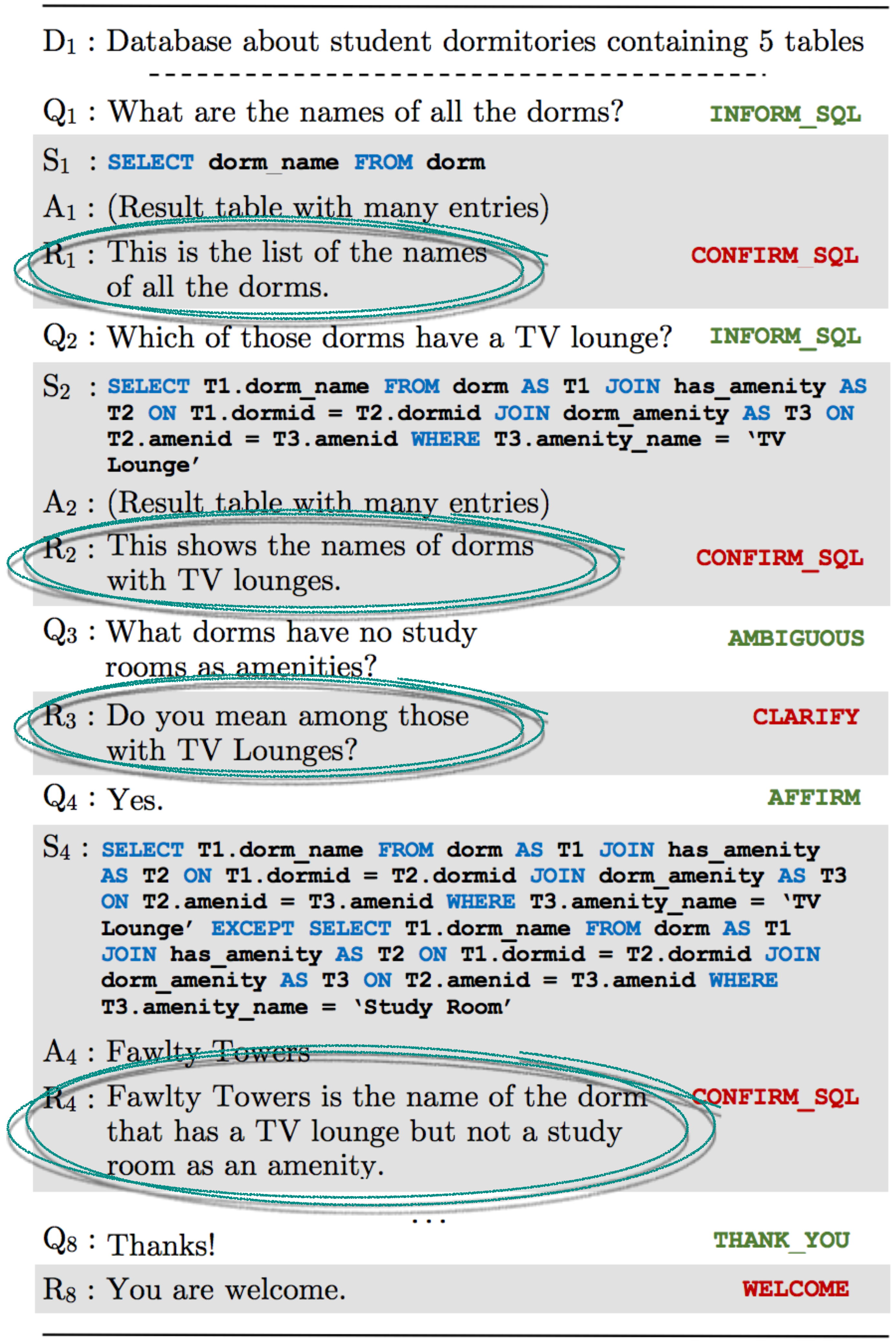
In real world, users typically issue sequences of questions when querying a database

Context-dependent utterances reflect special linguistic phenomena such as co-references and omission

Besides well-formed **information seeking** questions, users may issue utterances that require **clarification** and trigger **other dialogue actions**

System responses are better to be paired w/ accessible **natural language responses**





D_1 :	Database about student dormitories containing 5 tables	
Q_1 :	What are the names of all the dorms?	INFORM_SQL
S_1 :	`SELECT dorm_name FROM dorm`	
A_1 :	(Result table with many entries)	
R_1 :	This is the list of the names of all the dorms.	CONFIRM_SQL
Q_2 :	Which of those dorms have a TV lounge?	INFORM_SQL
S_2 :	`SELECT T1.dorm_name FROM dorm AS T1 JOIN has_amenity AS T2 ON T1.dormid = T2.dormid JOIN dorm_amenity AS T3 ON T2.amenid = T3.amenid WHERE T3.amenity_name = ‘TV Lounge’`	
A_2 :	(Result table with many entries)	
R_2 :	This shows the names of dorms with TV lounges.	CONFIRM_SQL
Q_3 :	What dorms have no study rooms as amenities?	AMBIGUOUS
R_3 :	Do you mean among those with TV Lounges?	CLARIFY
Q_4 :	Yes.	AFFIRM
S_4 :	`SELECT T1.dorm_name FROM dorm AS T1 JOIN has_amenity AS T2 ON T1.dormid = T2.dormid JOIN dorm_amenity AS T3 ON T2.amenid = T3.amenid WHERE T3.amenity_name = ‘TV Lounge’ EXCEPT SELECT T1.dorm_name FROM dorm AS T1 JOIN has_amenity AS T2 ON T1.dormid = T2.dormid JOIN dorm_amenity AS T3 ON T2.amenid = T3.amenid WHERE T3.amenity_name = ‘Study Room’`	
A_4 :	Fawlty Towers	
R_4 :	Fawlty Towers is the name of the dorm that has a TV lounge but not a study room as an amenity.	CONFIRM_SQL
	…	
Q_8 :	Thanks!	THANK_YOU
R_8 :	You are welcome.	WELCOME

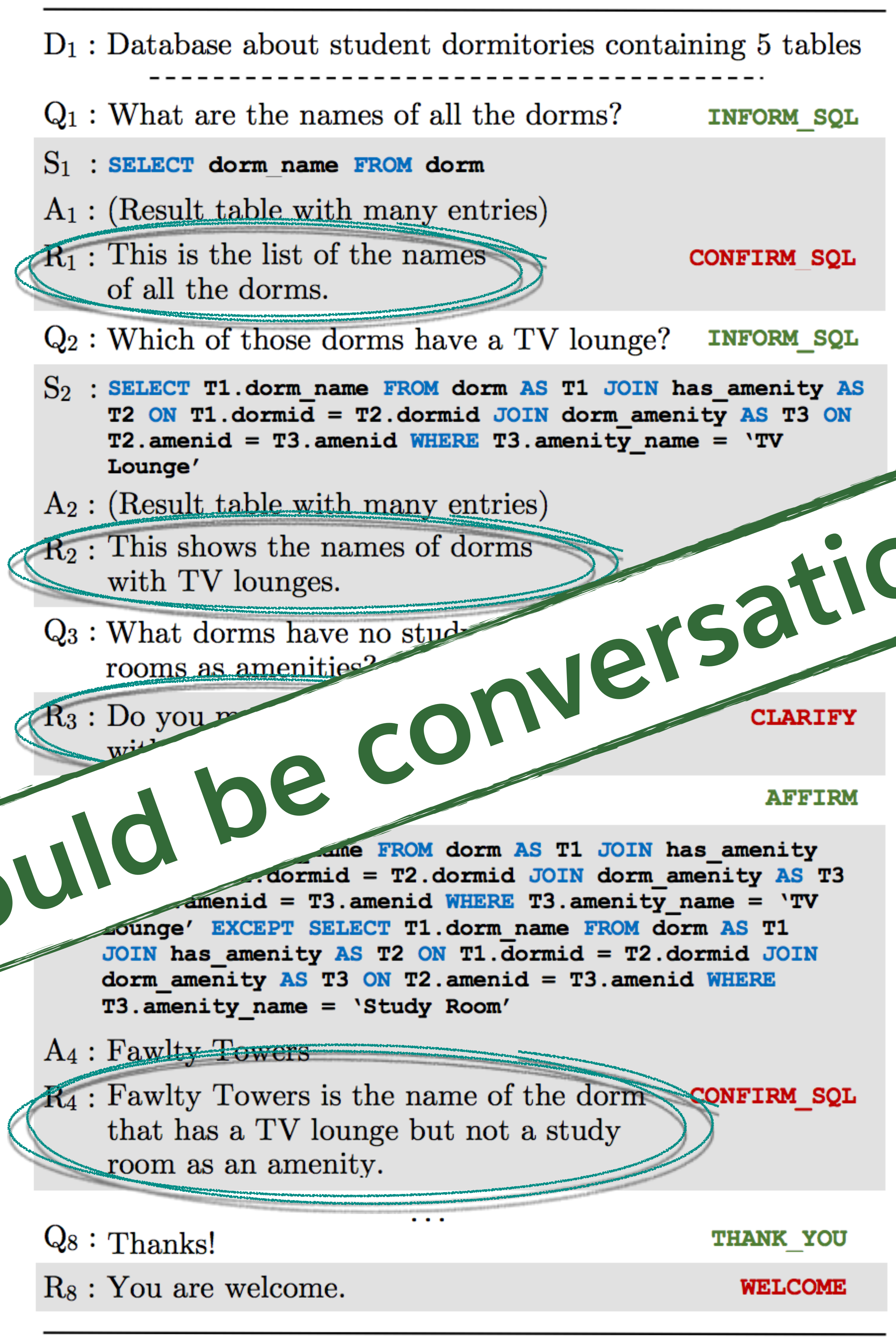
CoSQL: A Conversational Text-to-SQL Challenge

In real world, users typically issue sequences of questions when querying a database

Context-dependent utterances reflect special linguistic phenomena such as co-references and omission

Besides well-formed **information seeking** questions, users may issue utterances that require **clarification** and trigger **other dialogue actions**

System responses are better to be paired w/ accessib **natural language responses**



D_1 : Database about student dormitories containing 5 tables

Q_1 : What are the names of all the dorms? INFORM_SQL

S_1 : `SELECT dorm_name FROM dorm`

A_1 : (Result table with many entries)

R_1 : This is the list of the names of all the dorms. CONFIRM_SQL

Q_2 : Which of those dorms have a TV lounge? INFORM_SQL

S_2 :

```
SELECT T1.dorm_name FROM dorm AS T1 JOIN has_amenity AS
T2 ON T1.dormid = T2.dormid JOIN dorm_amenity AS T3 ON
T2.amenid = T3.amenid WHERE T3.amenity_name = 'TV
Lounge'
```

A_2 : (Result table with many entries)

R_2 : This shows the names of dorms with TV lounges.

Q_3 : What dorms have no stud rooms as amenities?

R_3 : Do you m wi CLARIFY

AFFIRM

```
me FROM dorm AS T1 JOIN has_amenity
.dormid = T2.dormid JOIN dorm_amenity AS T3
amenid = T3.amenid WHERE T3.amenity_name = 'TV
ounge' EXCEPT SELECT T1.dorm_name FROM dorm AS T1
JOIN has_amenity AS T2 ON T1.dormid = T2.dormid JOIN
dorm_amenity AS T3 ON T2.amenid = T3.amenid WHERE
T3.amenity_name = 'Study Room'
```

A_4 : Fawlty Towers

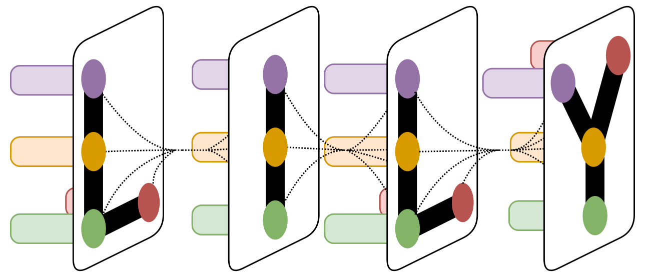
R_4 : Fawlty Towers is the name of the dorm that has a TV lounge but not a study room as an amenity. CONFIRM_SQL

...

Q_8 : Thanks! THANK_YOU

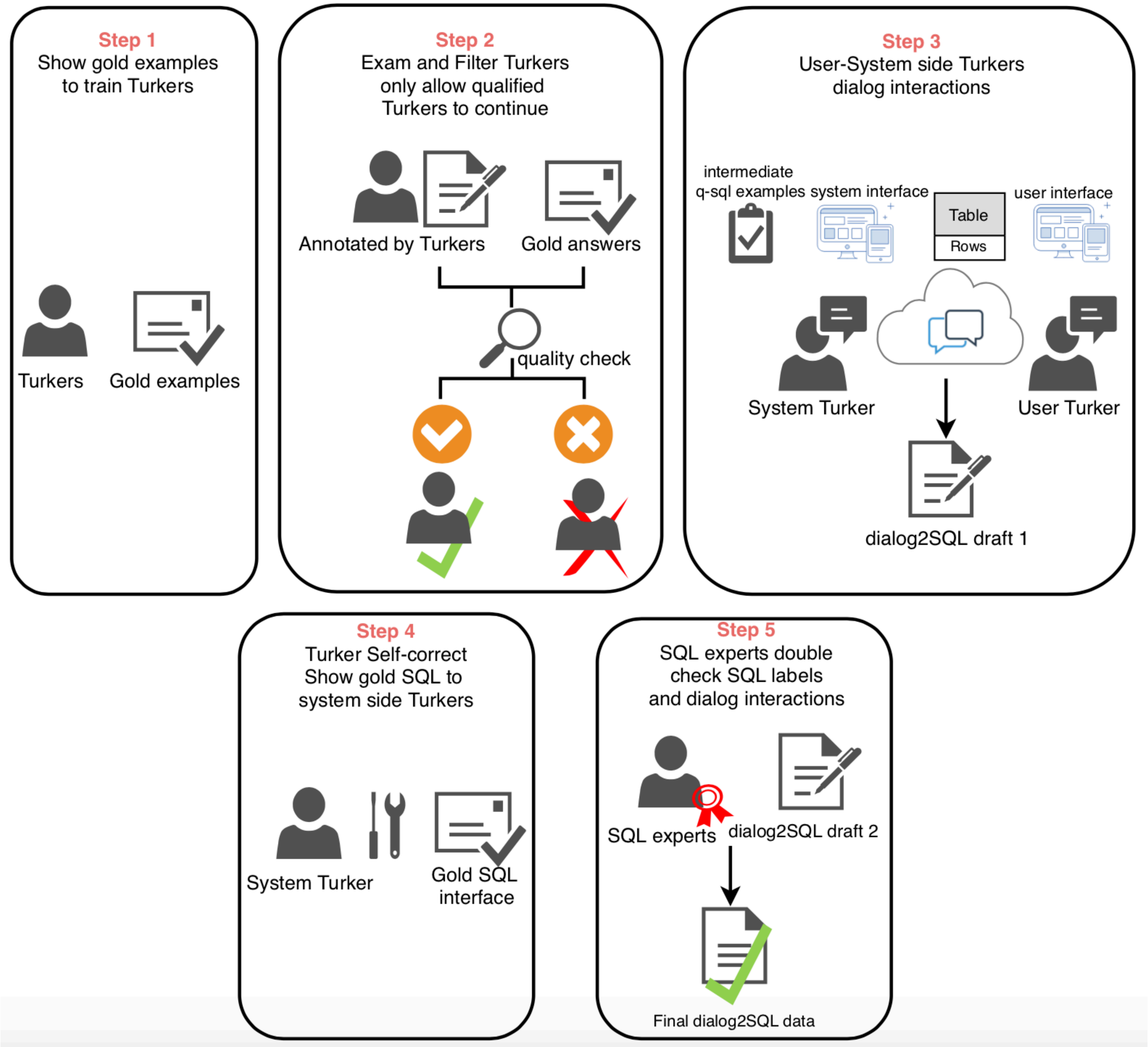
R_8 : You are welcome. WELCOME

NLIDB should be conversational



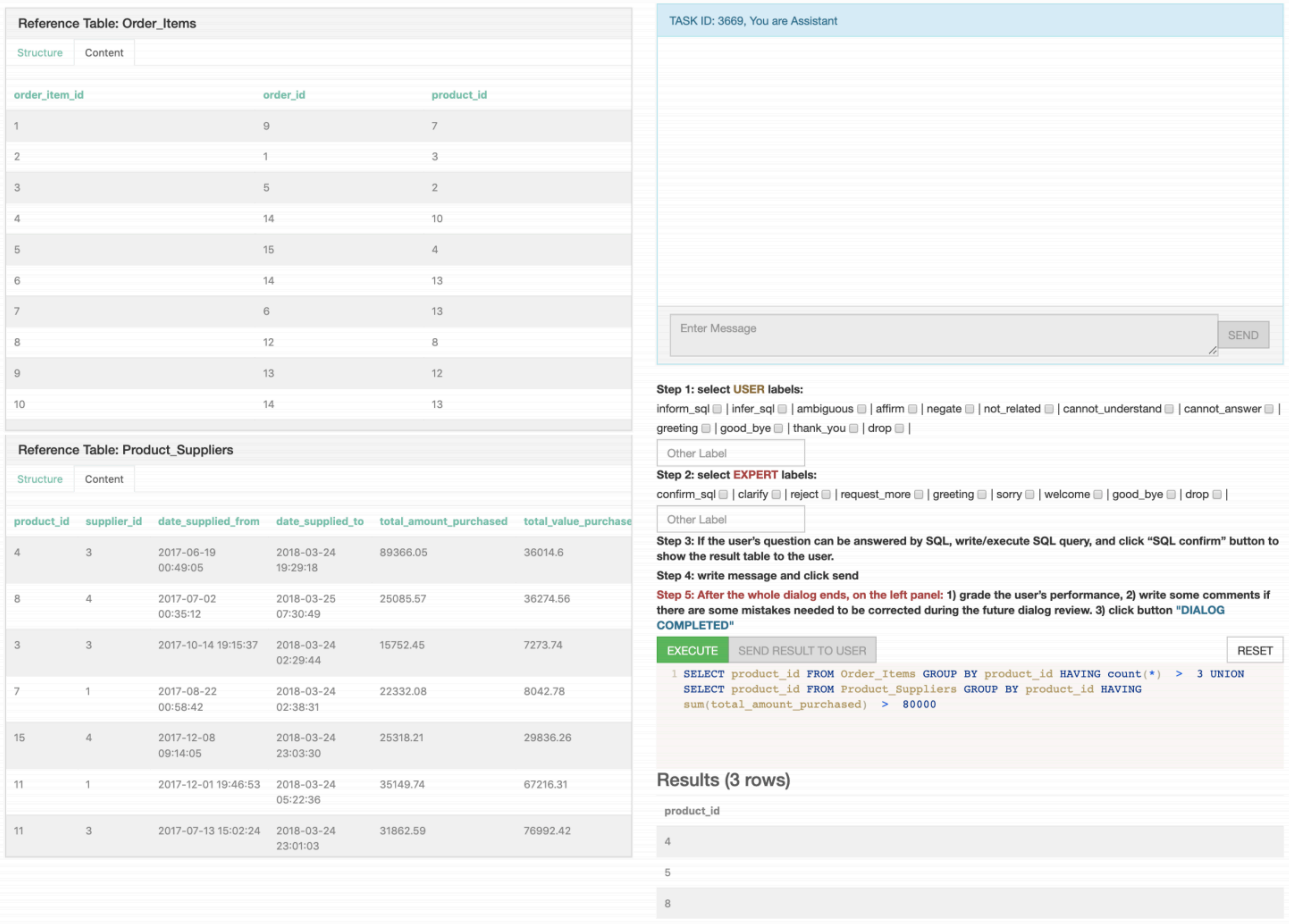
Wizard-of-Oz Data Collection

Wizard-of-Oz data collection pipeline:



Step 1
Show gold examples to train Turkers
Turkers
Gold examples
Step 2
Exam and Filter Turkers only allow qualified Turkers to continue
Annotated by Turkers
Gold answers
quality check
Step 3
User-System side Turkers dialog interactions
intermediate q-sql examples
system interface
Table
Rows
user interface
System Turker
User Turker
dialog2SQL draft 1
Step 4
Turker Self-correct Show gold SQL to system side Turkers
System Turker
Gold SQL interface
Step 5
SQL experts double check SQL labels and dialog interactions
SQL experts
dialog2SQL draft 2
Final dialog2SQL data

Chatting Interface



Reference Table: Order_Items

Structure | Content

order_item_id	order_id	product_id
1	9	7
2	1	3
3	5	2
4	14	10
5	15	4
6	14	13
7	6	13
8	12	8
9	13	12
10	14	13

Reference Table: Product_Suppliers

Structure | Content

product_id	supplier_id	date_supplied_from	date_supplied_to	total_amount_purchased	total_value_purchase
4	3	2017-06-19 00:49:05	2018-03-24 19:29:18	89366.05	36014.6
8	4	2017-07-02 00:35:12	2018-03-25 07:30:49	25085.57	36274.56
3	3	2017-10-14 19:15:37	2018-03-24 02:29:44	15752.45	7273.74
7	1	2017-08-22 00:58:42	2018-03-24 02:38:31	22332.08	8042.78
15	4	2017-12-08 09:14:05	2018-03-24 23:03:30	25318.21	29836.26
11	1	2017-12-01 19:46:53	2018-03-24 05:22:36	35149.74	67216.31
11	3	2017-07-13 15:02:24	2018-03-24 23:01:03	31862.59	76992.42

TASK ID: 3669, You are Assistant

Enter Message

SEND

Step 1: select USER labels:

inform_sql □ | infer_sql □ | ambiguous □ | affirm □ | negate □ | not_related □ | cannot_understand □ | cannot_answer □ | greeting □ | good_bye □ | thank_you □ | drop □ |

Other Label

Step 2: select EXPERT labels:

confirm_sql □ | clarify □ | reject □ | request_more □ | greeting □ | sorry □ | welcome □ | good_bye □ | drop □ |

Other Label

Step 3: If the user’s question can be answered by SQL, write/execute SQL query, and click “SQL confirm” button to show the result table to the user.

Step 4: write message and click send

Step 5: After the whole dialog ends, on the left panel: 1) grade the user's performance, 2) write some comments if there are some mistakes needed to be corrected during the future dialog review. 3) click button "DIALOG COMPLETED"

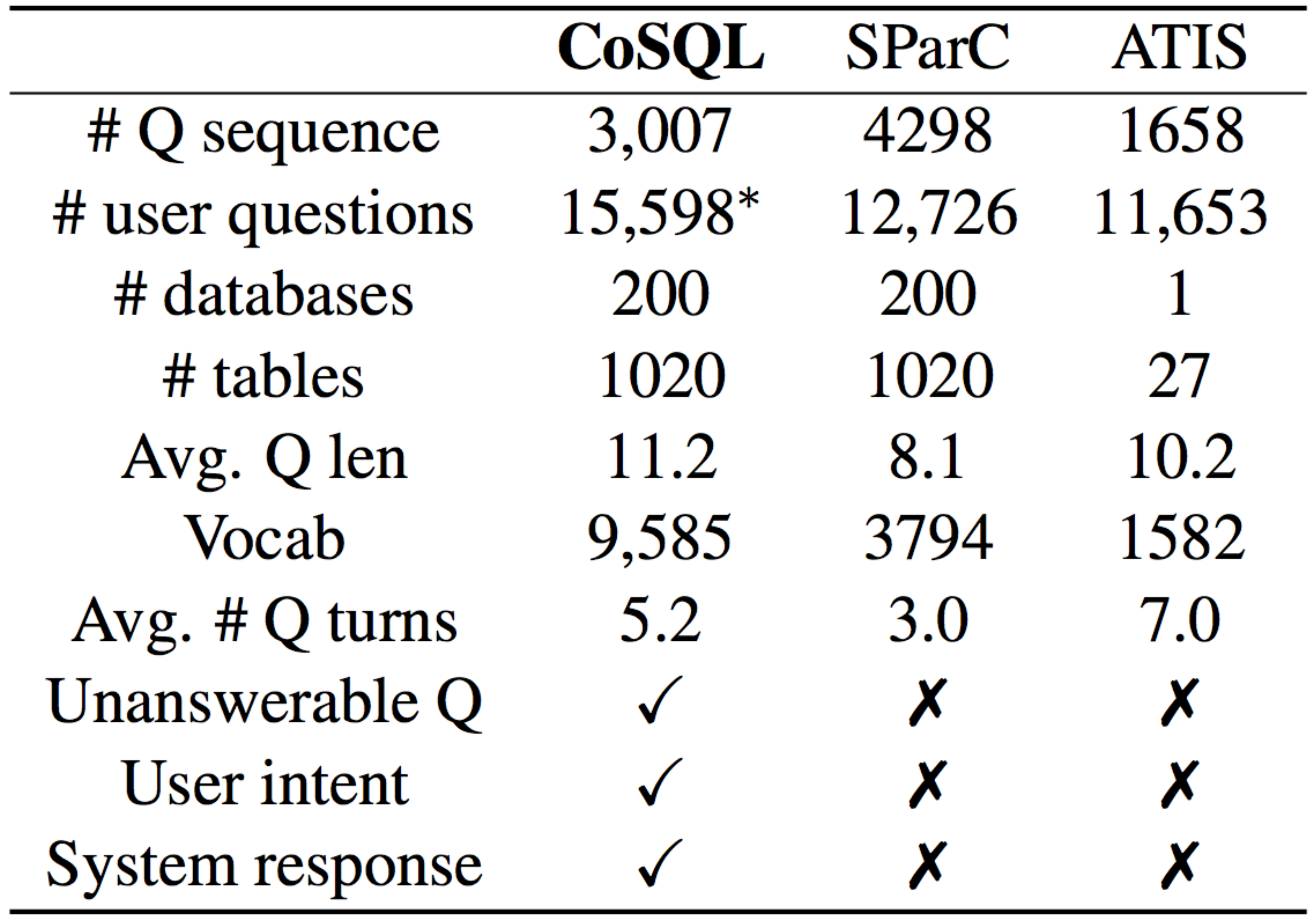
EXECUTE | SEND RESULT TO USER | RESET

```
SELECT product_id FROM Order_Items GROUP BY product_id HAVING count(*)  >  3 UNION
SELECT product_id FROM Product_Suppliers GROUP BY product_id HAVING
sum(total_amount_purchased)  >  80000
```

Results (3 rows)

product_id
4
5
8

Data Statistics



	CoSQL	SParC	ATIS
# Q sequence	3,007	4298	1658
# user questions	15,598*	12,726	11,653
# databases	200	200	1
# tables	1020	1020	27
Avg. Q len	11.2	8.1	10.2
Vocab	9,585	3794	1582
Avg. # Q turns	5.2	3.0	7.0
Unanswerable Q	✓	✗	✗
User intent	✓	✗	✗
System response	✓	✗	✗

Context-dependent text-to-SQL (I)

Natural language response generation (II)

Dialogue action prediction (III)

ATIS (Hemphill et al. 1990, Dahl et al. 1994); SParC (Yu et al. 2019)

Leaderboard: https://yale-lily.github.io/cosql

Yale
salesforce
M
UNIVERSITY OF MICHIGAN

CoSQL 1.0

A Conversational Text-to-SQL Challenge
Towards Cross-Domain Natural Language Interfaces to Databases

What is CoSQL?

CoSQL is a corpus for building cross-domain **Co**nversational text-to-**SQL** systems. It is the dilaogue version of the ***Spider*** and ***SParC*** tasks. CoSQL consists of 30k+ turns plus 10k+ annotated SQL queries, obtained from a Wizard-of-Oz collection of 3k dialogues querying 200 complex databases spanning 138 domains. Each dialogue simulates a real-world DB query scenario with a crowd worker as a user exploring the database and a SQL expert retrieving answers with SQL, clarifying ambiguous questions, or otherwise informing of unanswerable questions.

CoSQL Paper (EMNLP'19)

CoSQL Post

Leaderboard - SQL-grounded Dialogue State Tracking

In CoSQL, user dialogue states are grounded in SQL queries. Dialogue state tracking (DST) in this case is to predict the correct SQL query for each user utterance with `INFORM_SQL` label given the interaction context and the DB schema. Comparing to other context-dependent text-to-SQL tasks such as *SParC*, the DST task in CoSQL also includes the ambiguous questions if the user affirms the system clarification of them. In this case, the system clarification is also given as part of the interaction context to predict the SQL query corresponding to the question. As in *Spider* and *SParC* tasks, we report results of Exact Set Match without Values here.

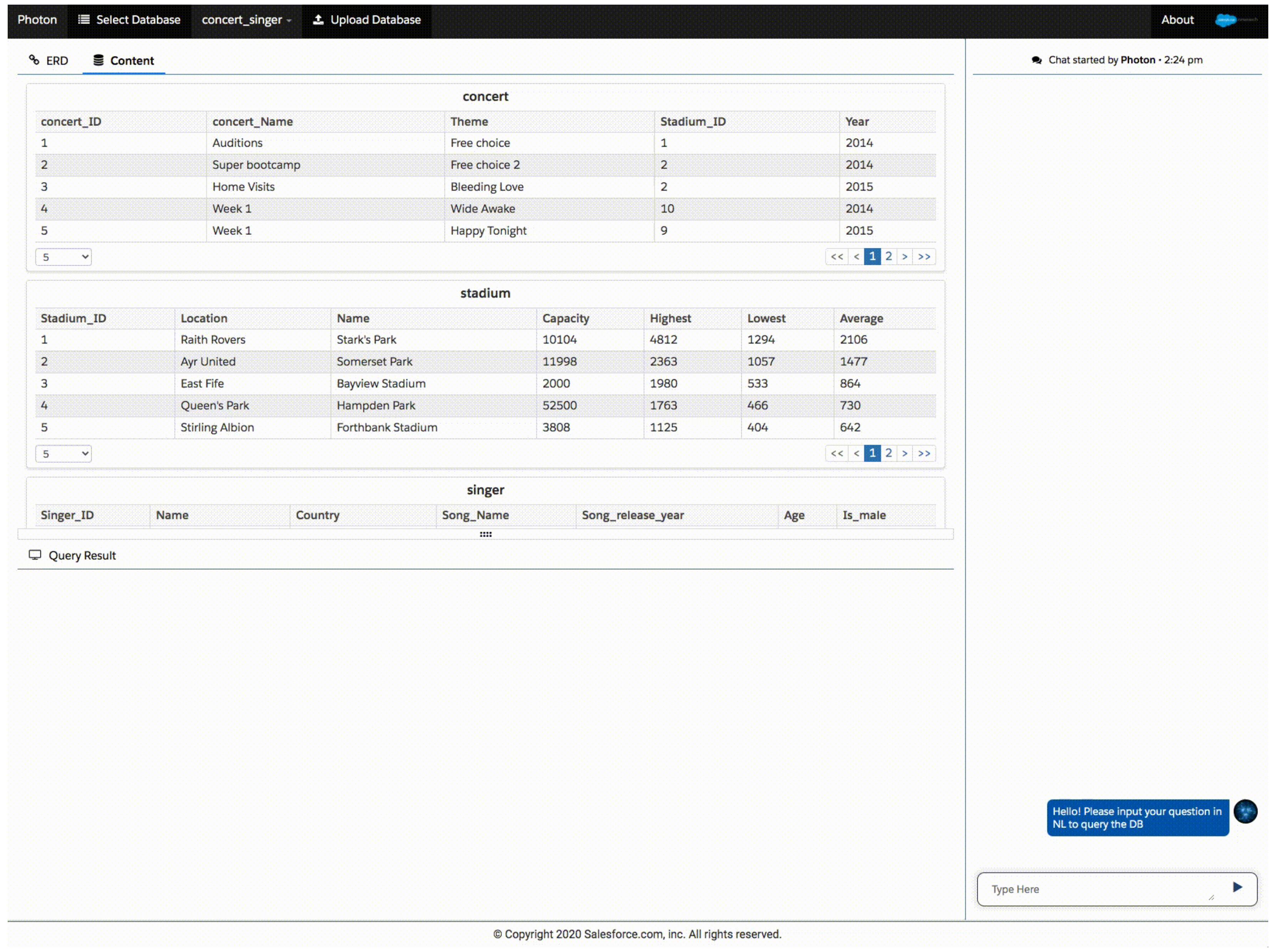
Rank	Model	Question Match	Interaction Match
1 Aug 30, 2019	EditSQL *Yale University & Salesforce Research* (Zhang et al. EMNLP '19) code	**40.8**	**13.7**

I. Content-Aware Textual-Tabular Encodings for Table Semantic Parsing (TSP)

Bridging Textual and Tabular Data for Cross-Domain Text-to-SQL Semantic Parsing. Lin et al. 2020.

Live Demo: https://naturalsql.com

Open Source: https://github.com/salesforce/TabularSemanticParsing



concert

concert_ID	concert_Name	Theme	Stadium_ID	Year
1	Auditions	Free choice	1	2014
2	Super bootcamp	Free choice 2	2	2014
3	Home Visits	Bleeding Love	2	2015
4	Week 1	Wide Awake	10	2014
5	Week 1	Happy Tonight	9	2015

stadium

Stadium_ID	Location	Name	Capacity	Highest	Lowest	Average
1	Raith Rovers	Stark's Park	10104	4812	1294	2106
2	Ayr United	Somerset Park	11998	2363	1057	1477
3	East Fife	Bayview Stadium	2000	1980	533	864
4	Queen's Park	Hampden Park	52500	1763	466	730
5	Stirling Albion	Forthbank Stadium	3808	1125	404	642

singer

Singer_ID	Name	Country	Song_Name	Song_release_year	Age	Is_male

Chat started by **Photon** · 2:24 pm

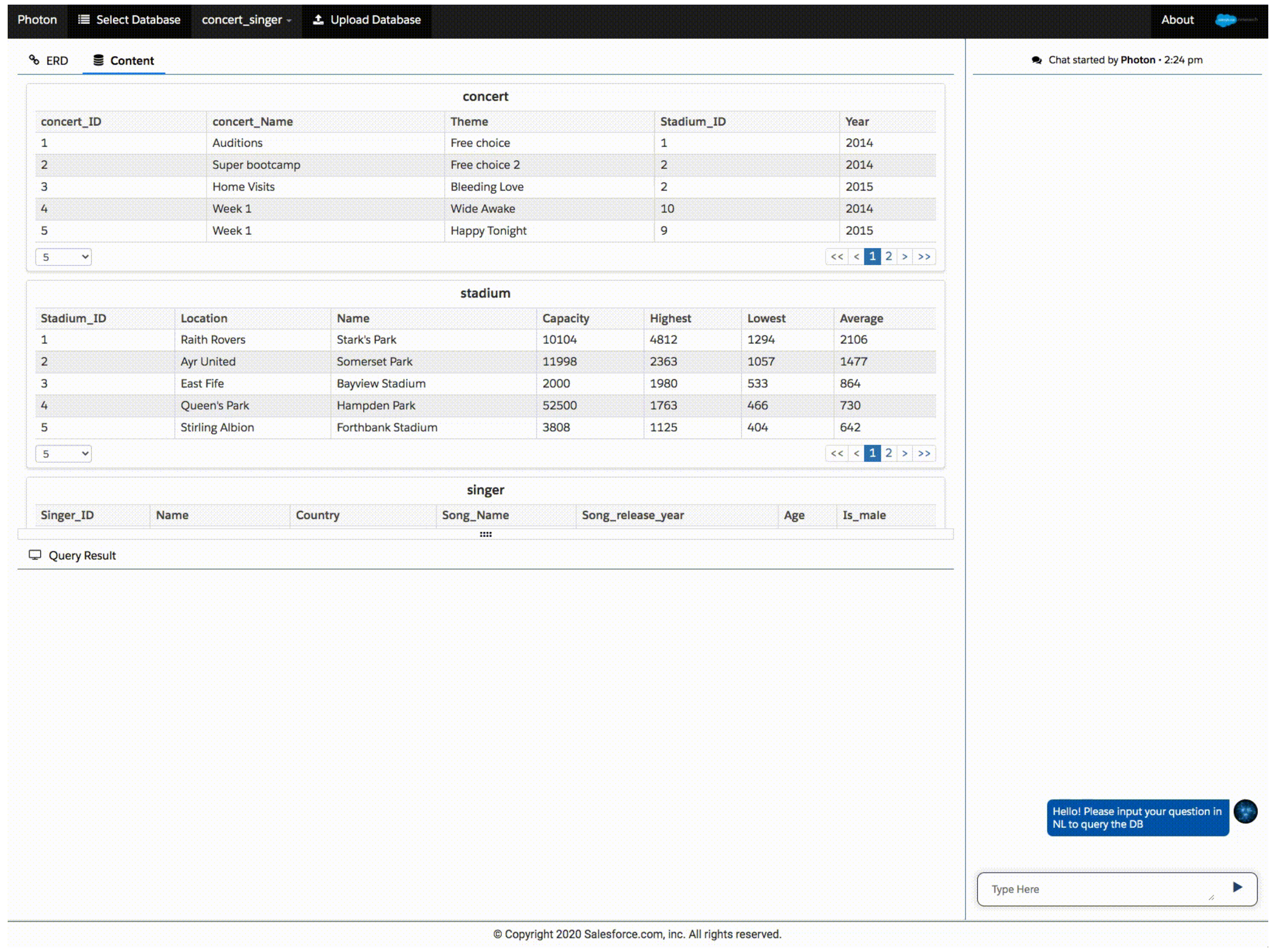
Hello! Please input your question in NL to query the DB

I. Content-Aware Textual-Tabular Encodings for Table Semantic Parsing (TSP)

Bridging Textual and Tabular Data for Cross-Domain Text-to-SQL Semantic Parsing. Lin et al. 2020.

Live Demo: https://naturalsql.com

Open Source: https://github.com/salesforce/TabularSemanticParsing



Chat started by **Photon** • 2:24 pm

concert

concert_ID	concert_Name	Theme	Stadium_ID	Year
1	Auditions	Free choice	1	2014
2	Super bootcamp	Free choice 2	2	2014
3	Home Visits	Bleeding Love	2	2015
4	Week 1	Wide Awake	10	2014
5	Week 1	Happy Tonight	9	2015

5 << < 1 2 > >>

stadium

Stadium_ID	Location	Name	Capacity	Highest	Lowest	Average
1	Raith Rovers	Stark's Park	10104	4812	1294	2106
2	Ayr United	Somerset Park	11998	2363	1057	1477
3	East Fife	Bayview Stadium	2000	1980	533	864
4	Queen's Park	Hampden Park	52500	1763	466	730
5	Stirling Albion	Forthbank Stadium	3808	1125	404	642

5 << < 1 2 > >>

singer

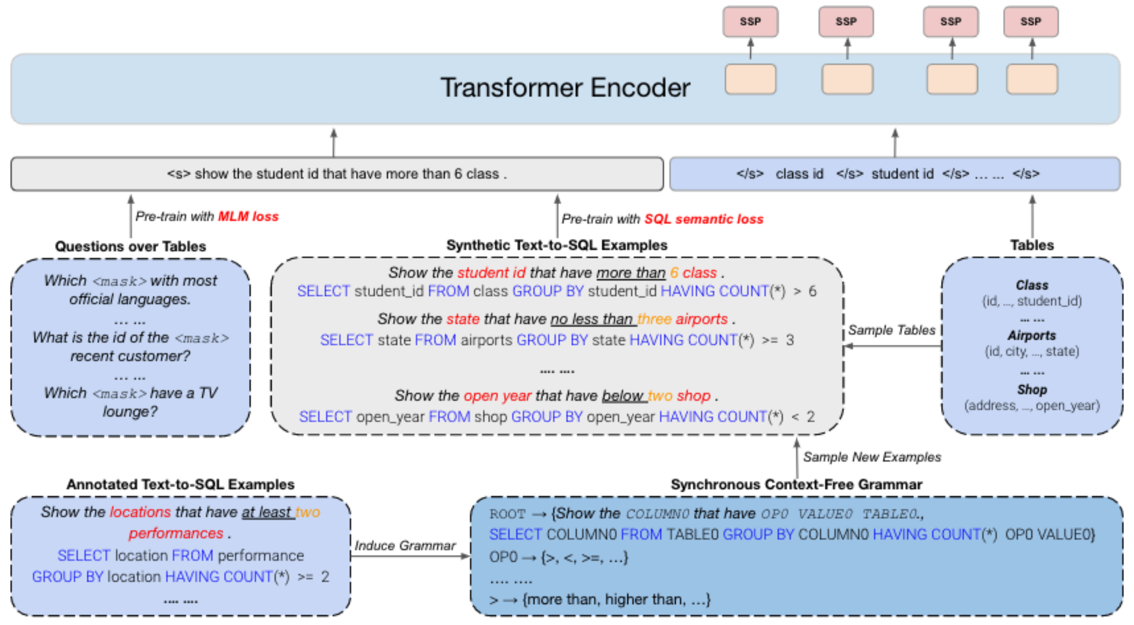
Singer_ID	Name	Country	Song_Name	Song_release_year	Age	Is_male

Hello! Please input your question in NL to query the DB

© Copyright 2020 Salesforce.com, inc. All rights reserved.

II. Pre-training Textual-Tabular Representations with Semantic Scaffolds

GraPPa: Grammar-Augmented Pre-training for Table Semantic Parsing. Yu et al. 2020.


SSP
SSP
SSP
SSP
Transformer Encoder
<s> show the student id that have more than 6 class .
</s> class id </s> student id </s> ... ... </s>
Pre-train with MLM loss
Pre-train with SQL semantic loss
Questions over Tables
Synthetic Text-to-SQL Examples
Tables
Sample Tables
Sample New Examples
Annotated Text-to-SQL Examples
Induce Grammar
Synchronous Context-Free Grammar

Which <mask> with most official languages.
... ...
What is the id of the <mask> recent customer?
... ...
Which <mask> have a TV lounge?

Show the student id that have more than 6 class .
SELECT student_id FROM class GROUP BY student_id HAVING COUNT(*) > 6
Show the state that have no less than three airports .
SELECT state FROM airports GROUP BY state HAVING COUNT(*) >= 3
....
Show the open year that have below two shop .
SELECT open_year FROM shop GROUP BY open_year HAVING COUNT(*) < 2

Class (id, ..., student_id)
... ...
Airports (id, city, ..., state)
... ...
Shop (address, ..., open_year)

Show the locations that have at least two performances .
SELECT location FROM performance GROUP BY location HAVING COUNT(*) >= 2
....

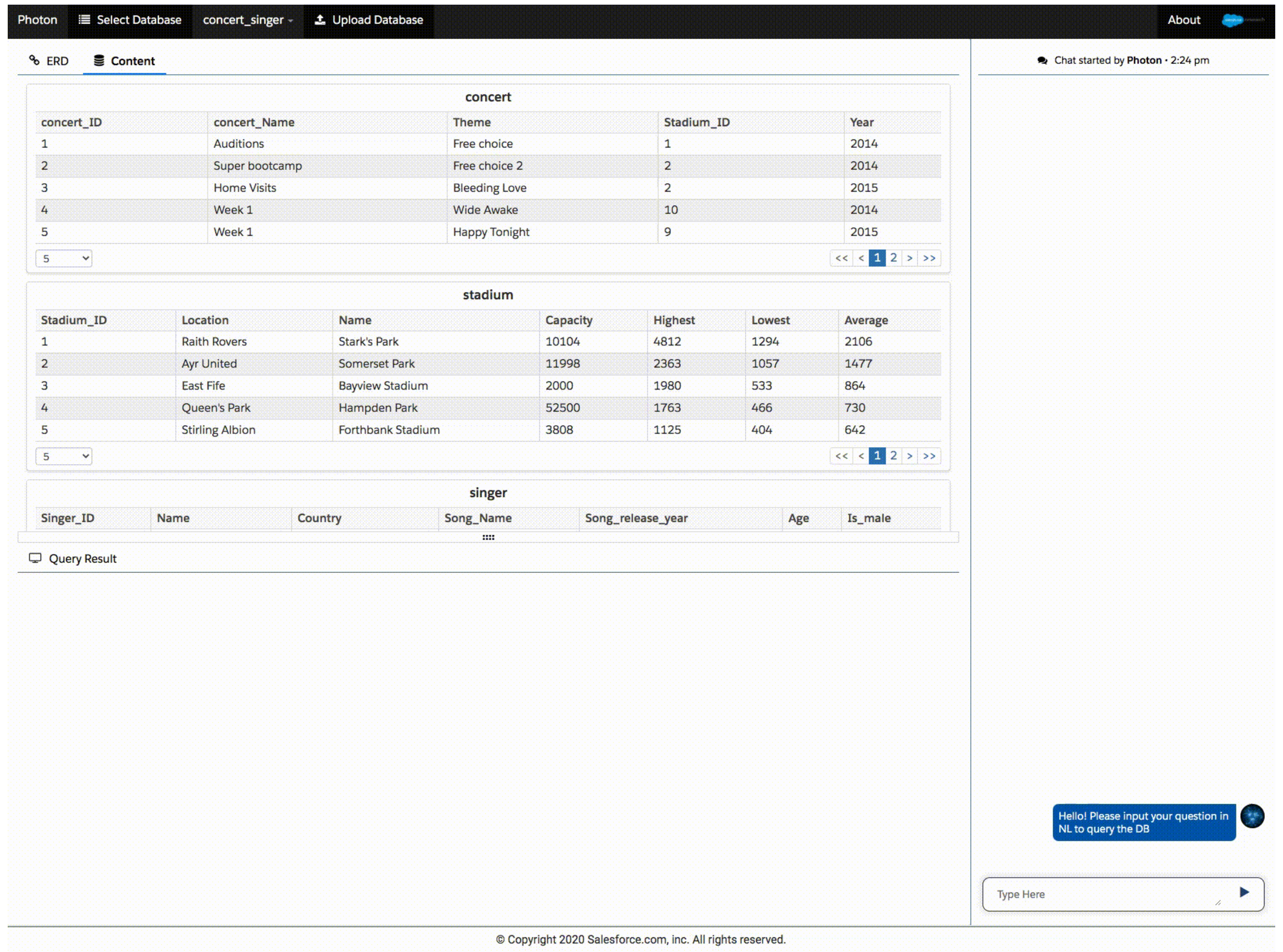
ROOT → {Show the COLUMN0 that have OP0 VALUE0 TABLE0.,
SELECT COLUMN0 FROM TABLE0 GROUP BY COLUMN0 HAVING COUNT(*) OP0 VALUE0}
OP0 → {>, <, >=, ...}
....
> → {more than, higher than, ...}

I. Content-Aware Textual-Tabular Encodings for Table Semantic Parsing (TSP)

Bridging Textual and Tabular Data for Cross-Domain Text-to-SQL Semantic Parsing. Lin et al. 2020.

Live Demo: https://naturalsql.com

Open Source: https://github.com/salesforce/TabularSemanticParsing



concert

concert_ID	concert_Name	Theme	Stadium_ID	Year
1	Auditions	Free choice	1	2014
2	Super bootcamp	Free choice 2	2	2014
3	Home Visits	Bleeding Love	2	2015
4	Week 1	Wide Awake	10	2014
5	Week 1	Happy Tonight	9	2015

stadium

Stadium_ID	Location	Name	Capacity	Highest	Lowest	Average
1	Raith Rovers	Stark's Park	10104	4812	1294	2106
2	Ayr United	Somerset Park	11998	2363	1057	1477
3	East Fife	Bayview Stadium	2000	1980	533	864
4	Queen's Park	Hampden Park	52500	1763	466	730
5	Stirling Albion	Forthbank Stadium	3808	1125	404	642

singer

Singer_ID	Name	Country	Song_Name	Song_release_year	Age	Is_male

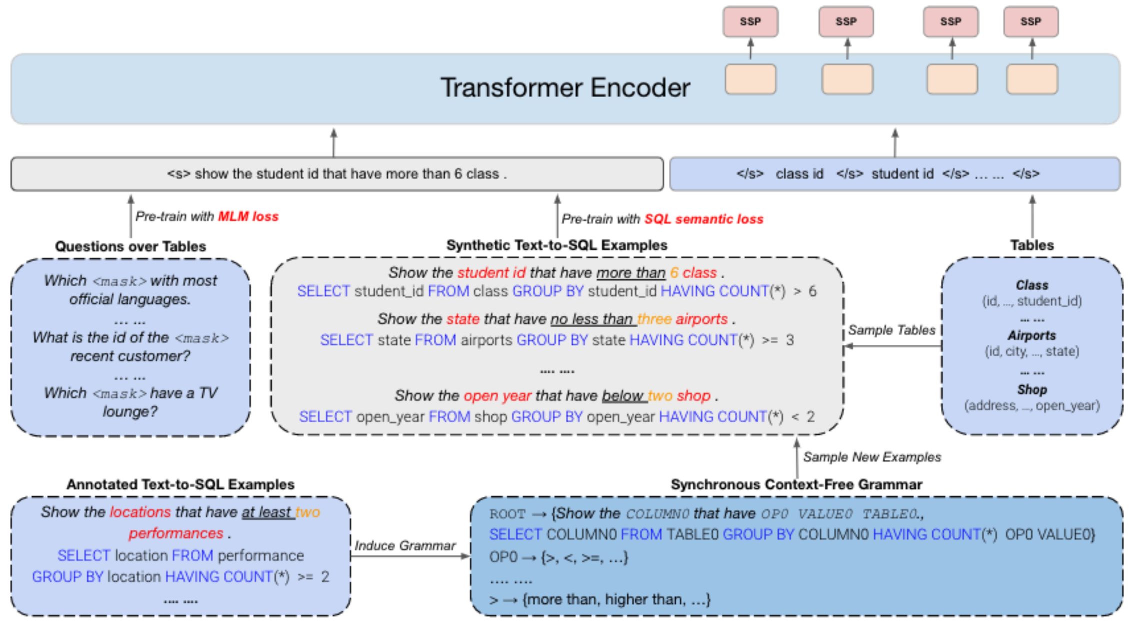
Chat started by **Photon** · 2:24 pm

Hello! Please input your question in NL to query the DB

© Copyright 2020 Salesforce.com, inc. All rights reserved.

II. Pre-training Textual-Tabular Representations with Semantic Scaffolds

GraPPa: Grammar-Augmented Pre-training for Table Semantic Parsing. Yu et al. 2020.


SSP
SSP
SSP
SSP
Transformer Encoder
<s> show the student id that have more than 6 class .
</s> class id </s> student id </s> ... ... </s>
Pre-train with MLM loss
Pre-train with SQL semantic loss
Questions over Tables
Synthetic Text-to-SQL Examples
Tables
Sample Tables
Sample New Examples
Annotated Text-to-SQL Examples
Induce Grammar
Synchronous Context-Free Grammar

Which <mask> with most official languages.
... ...
What is the id of the <mask> recent customer?
... ...
Which <mask> have a TV lounge?

Show the student id that have more than 6 class .
SELECT student_id FROM class GROUP BY student_id HAVING COUNT(*) > 6
Show the state that have no less than three airports .
SELECT state FROM airports GROUP BY state HAVING COUNT(*) >= 3
....
Show the open year that have below two shop .
SELECT open_year FROM shop GROUP BY open_year HAVING COUNT(*) < 2

Class (id, ..., student_id)
... ...
Airports (id, city, ..., state)
... ...
Shop (address, ..., open_year)

Show the locations that have at least two performances .
SELECT location FROM performance GROUP BY location HAVING COUNT(*) >= 2
....

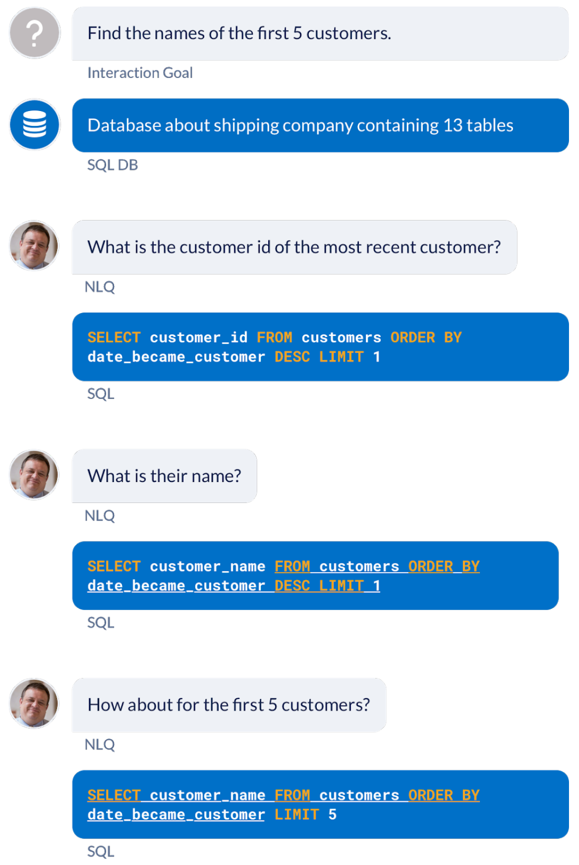
ROOT → {Show the COLUMN0 that have OP0 VALUE0 TABLE0.,
SELECT COLUMN0 FROM TABLE0 GROUP BY COLUMN0 HAVING COUNT(*) OP0 VALUE0}
OP0 → {>, <, >=, ...}
....
> → {more than, higher than, ...}

III. Conversational Table Semantic Parsing

SParC: Cross-Domain Semantic Parsing in Context. Yu et al. 2019.

Editing-Based SQL Query Generation for Cross-Domain Context-Dependent Questions. Zhang et al. 2019.

CoSQL: A Conversational Text-to-SQL Challenge Towards Cross-Domain Natural Language Interfaces to Databases. Yu et al. 2019.



Find the names of the first 5 customers.
Interaction Goal

Database about shipping company containing 13 tables
SQL DB

What is the customer id of the most recent customer?
NLQ

```
SELECT customer_id FROM customers ORDER BY
date_became_customer DESC LIMIT 1
```
SQL

What is their name?
NLQ

```
SELECT customer_name FROM customers ORDER BY
date_became_customer DESC LIMIT 1
```
SQL

How about for the first 5 customers?
NLQ

```
SELECT customer_name FROM customers ORDER BY
date_became_customer LIMIT 5
```
SQL

thank you
BLAZE YOUR TRAIL
salesforce