Business School
UNIVERSITY OF COLORADO DENVER

Information Systems Program

Module 6
SQL for Data Mining Input

Lesson 7: Overview of SQL for Training Data with Top Event History

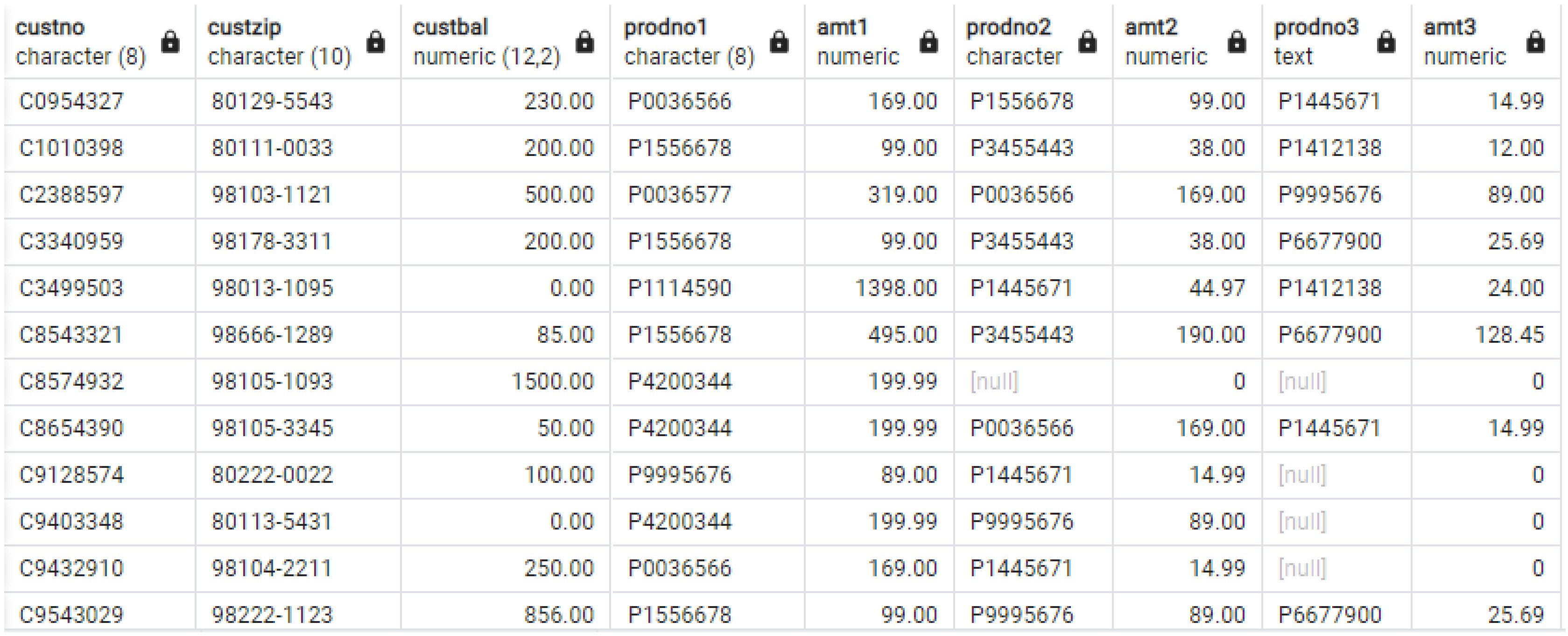
Lesson Objectives

- Explain input format for limited event history
- Explain statement pattern for the combined query
- Explain statement pattern for the common table expression part of the complete query

Input Format for Top Event History

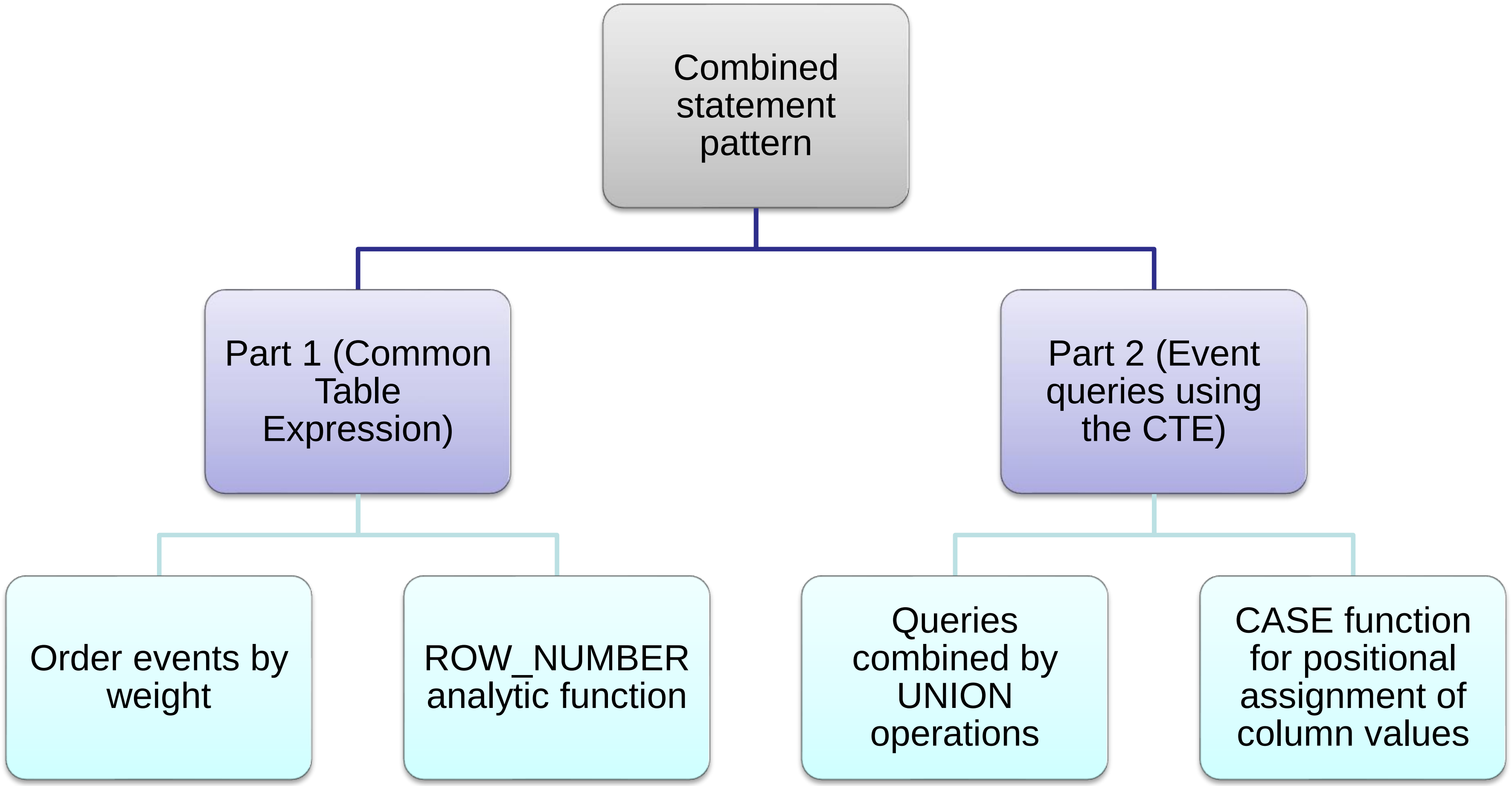
- **Entity variables**
  - Object to classify
  - For customer, use location, age, and income level
- **Event history**
  - Set of *n* event pairs <EventId, EventWeight>
  - Position ordered by event weight
  - Uniform versus non uniform with missing values

Input Format Example



custno character (8)	custzip character (10)	custbal numeric (12,2)	prodno1 character (8)	amt1 numeric	prodno2 character	amt2 numeric	prodno3 text	amt3 numeric
C0954327	80129-5543	230.00	P0036566	169.00	P1556678	99.00	P1445671	14.99
C1010398	80111-0033	200.00	P1556678	99.00	P3455443	38.00	P1412138	12.00
C2388597	98103-1121	500.00	P0036577	319.00	P0036566	169.00	P9995676	89.00
C3340959	98178-3311	200.00	P1556678	99.00	P3455443	38.00	P6677900	25.69
C3499503	98013-1095	0.00	P1114590	1398.00	P1445671	44.97	P1412138	24.00
C8543321	98666-1289	85.00	P1556678	495.00	P3455443	190.00	P6677900	128.45
C8574932	98105-1093	1500.00	P4200344	199.99	[null]	0	[null]	0
C8654390	98105-3345	50.00	P4200344	199.99	P0036566	169.00	P1445671	14.99
C9128574	80222-0022	100.00	P9995676	89.00	P1445671	14.99	[null]	0
C9403348	80113-5431	0.00	P4200344	199.99	P9995676	89.00	[null]	0
C9432910	98104-2211	250.00	P0036566	169.00	P1445671	14.99	[null]	0
C9543029	98222-1123	856.00	P1556678	99.00	P9995676	89.00	P6677900	25.69

Overview of the Statement Pattern (Non-Uniform Event History)


Combined statement pattern
Part 1 (Common Table Expression)
Part 2 (Event queries using the CTE)
Order events by weight
ROW_NUMBER analytic function
Queries combined by UNION operations
CASE function for positional assignment of column values

Statement Pattern for Non-Uniform History

```
-- Flatten nested events to N sets of event column pairs
CTE <CTEName> AS (
-- CTE provides an event ordering with ROW_NUMBER.
<CTESELECTStatment> )
-- Event queries retrieve event sizes 1 to N+
<EventQuery1>
UNION
<EventQuery2>
-- Event queries 2 to N+ use the CASE function.
…
UNION
<EventQueryN+> -- Only uses the top N events
[ ORDER BY <EntityIdColList> ]; -- optional ordering
```

Statement Pattern for the CTE Query

```
WITH <CTEName> AS (
-- Uses ROW_NUMBER analytic function to order events
SELECT <EntityCol1>, [ <EntityCol2>, …    <EntityColm>, ]
       <ItemIdCol>, <ItemWeightCol>,
       ROW_NUMBER() OVER ( PARTITION BY <EntityIdColList>
        ORDER BY <ItemWeightCol> [DESC] ) AS WeightRank
 FROM EntityTable, EventTable1, … [ EventTableN ETn ]
 WHERE <EntityTableJoinConditions>
 [ AND <EventTableJoinConditions> ]
 [ AND <TableConditions> ] )
-- <EntityIdColList> 1 column for data lake
-- Sometimes multiple columns for a data warehouse.
```

Summary

- Input format with entity variables and a fixed set of event pairs of event identifier and weight
- Positional assignment of event pair values
- Statement pattern with a CTE and UNION operations combining event queries
- ROW_NUMBER for ordering event pairs by event weight
- CASE function for positional assignment of values to event pairs