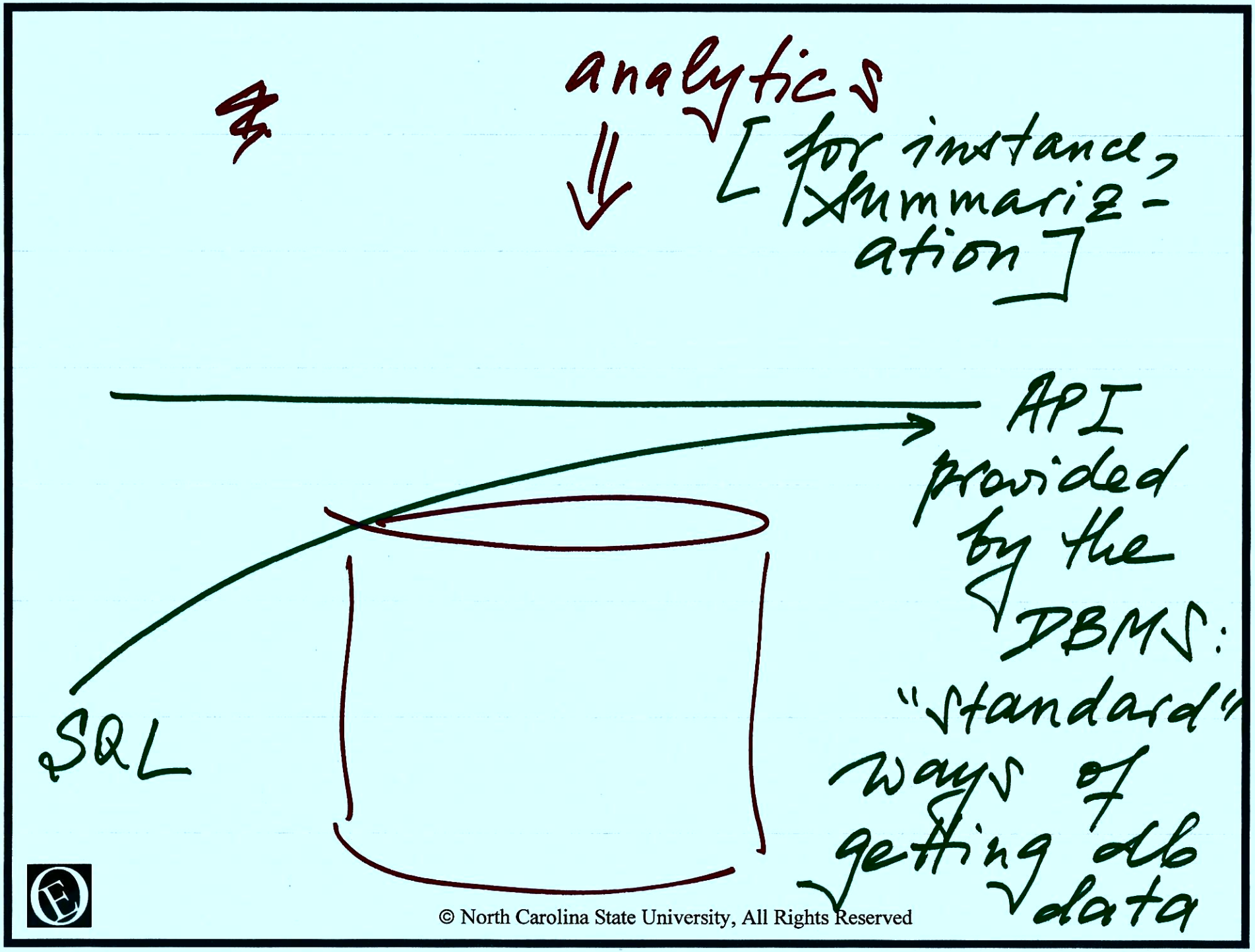
1. Design a db schema ⇒ relation schemas

2. Create schema (via SQL) ⇒ rel schemas captured in the DBMS

3. "Bulk load" initial data (via SQL) ⇒ stored relations with access through the DBMS APIs

4. Repeat: execute queries & data modifications (SQL)

© North Carolina State University, All Rights Reserved



© North Carolina State University, All Rights Reserved

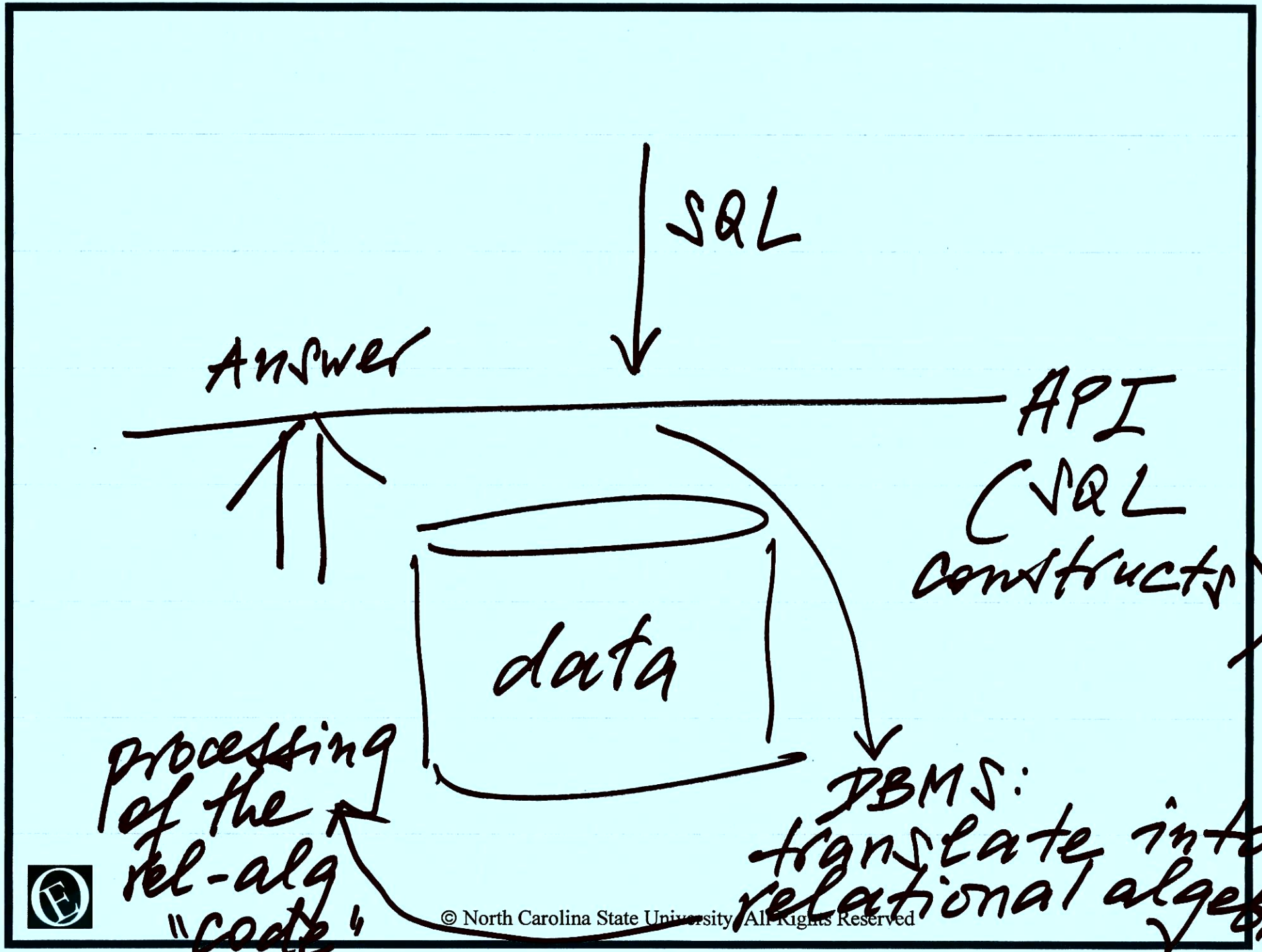
Standard representation of data (relations)

Standard API on the data

→ write code: procedural (relational algebra)

→ SQL: declarative

© North Carolina State University, All Rights Reserved



© North Carolina State University. All Rights Reserved

The Relational data model and "standard APIs"

- first in theory:
  - = relations
  - = and relational algebra
- then SQL came about (real life)

© North Carolina State University, All Rights Reserved

~~Elementary~~

Elementary operations on data:

("Take the fragment of the relation defined by a subset of its attributes")

– are composable: because many of them return relations

© North Carolina State University, All Rights Reserved

Going forward:

"schema of a relation"

means

"list of its attributes"

© North Carolina State University, All Rights Reserved

$R \cup S$, $R \cap S$, $R - S$ – not executable on given relations

R: | A | B |
|---|---|
| 1 | 2 |
| 3 | 4 |
| 5 | 6 |

S: | C | B |
|---|---|
| 1 | 2 |
| 3 | 4 |
| 7 | 8 |

T: | A | B |
|---|---|
| 5 | 6 |
| 7 | 8 |
| 2 | 1 |
| 1 | 2 |

These are executable:

$R \cup T$,
$R \cap T$,
$R - T$

– schema of output is same as schema of each input

© North Carolina State University, All Rights Reserved

RVT:

A	B
1	2
3	4
5	6
7	8
2	1

© North Carolina State University, All Rights Reserved

Renaming: $\rho(R_{HO})$

$\rho_{S(A,B)}(S) \Rightarrow$ S:

A	B
1	2
3	4
7	8

$R \cup (\rho_{S(A,B)}(V))$

© North Carolina State University, All Rights Reserved

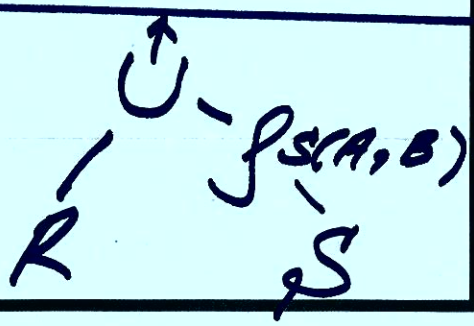
Ways of recording a relational-algebra expression:

A. $R \cup (\sigma_{\Psi(A,B)}(S))$ – single expression

B. As a sequence:

1. $R1 := \sigma_{\Psi(A,B)}(S)$
2. $R2 := R \cup R1$

C. Expression tree:



© North Carolina State University, All Rights Reserved

$f_{T(K,L)}(\nu) \Rightarrow$

T:

K	L
1	2
3	4
7	8

© North Carolina State University, All Rights Reserved

SQL:	Meaning in relational algebra
SELECT	← projection (π)
FROM	← cross product ($\times$)
WHERE	← selection (σ)

© North Carolina State University, All Rights Reserved

Operations in relational algebra (core) on sets of tuples

* Relation by name: R
* Set operators: $R \cup S$, $R \cap S$, $R - S$ . . . . . .
* Renaming: $\rho_{T(K,L)}(S)$
* Selection: $\sigma_{C}(R)$

C: Boolean expression: can use AND, OR, NOT

© North Carolina State University, All Rights Reserved

$\nabla_{GPA>3.7}(Student)$

Student: ID name addr GPA SA

traverse ⇒ ↓

.	.	.	3.5	.
.	.	.	(3.9)	.

© North Carolina State University, All Rights Reserved

* Projection:

$\pi_{\text{LIST OF ATTR NAMES}}(R)$

– duplicates in output are eliminated

U:

C	D	F
1	2	3
2	1	3
1	2	3

$\pi_{C,D}(U)$:

C	D
1	2
2	1

© North Carolina State University, All Rights Reserved

How do you eliminate duplicates in outcome of π:

√ Sorting: $n \log n$

C	D
1	2
2	1
1	2

$\Rightarrow\Rightarrow$

C	D
(√ sorted)	
1	2
1	2
2	1

$\Rightarrow$ eliminate

© North Carolina State University, All Rights Reserved

Example:

$\Pi_{GPA, SAT}(Student)$

Return IDs and dates of all the applications

$\Rightarrow$ in rel alg, duplicates are eliminated

– if you add "for all the students whose GPA > 3.7"

© North Carolina State University, All Rights Reserved