Lecture 22

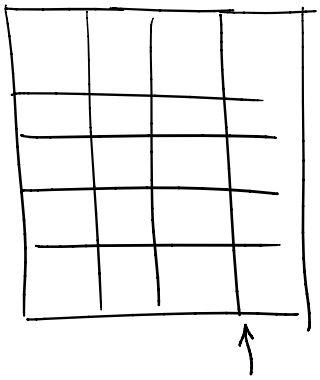
Tuesday, November 9, 2021 10:16 AM

Unit 3: Programming with Relations

SQL / Relational algebra

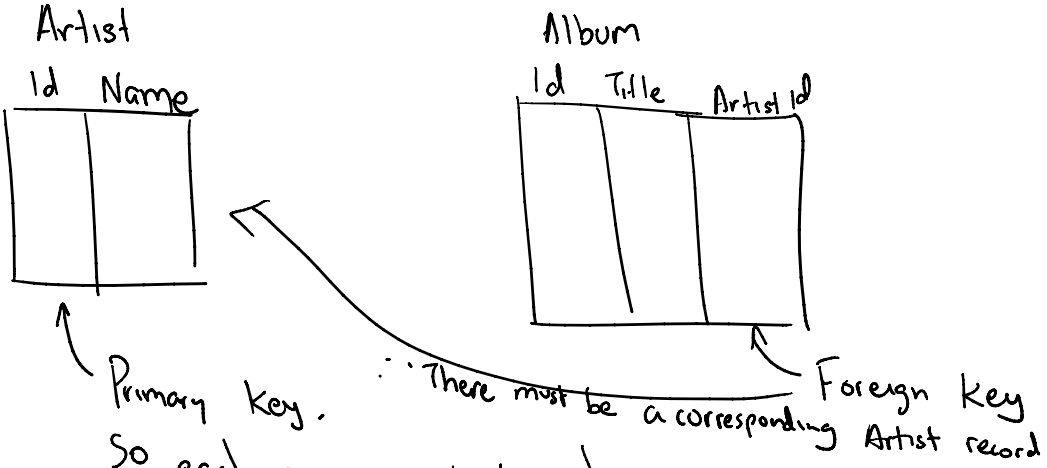
Table / Relation : set of tuples (multisets / bags) duplicates

~~order by~~



← Row / Tuples / Records

↑ Column / Fields / Attributes



SQL: Select, Insert, Update, Create table

↑

Drop table

, Update , Create table

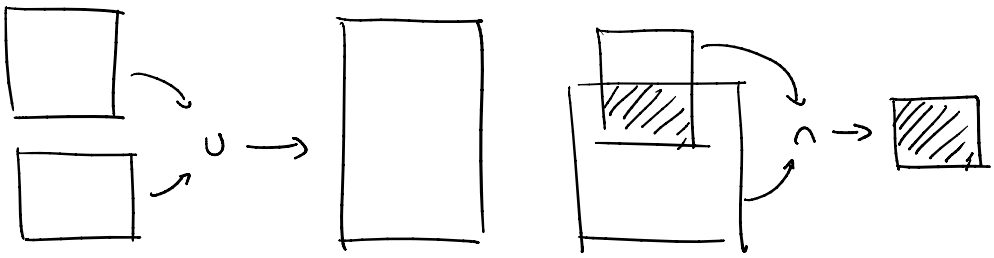
Drop table

(select _____

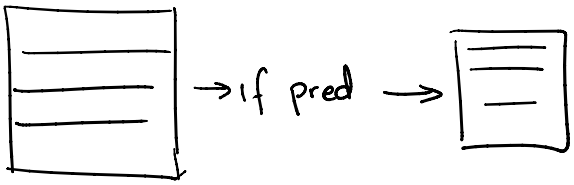
from $Table_1$, $Table_2$ Union

where _____) Intersects

Sets: Union Intersection

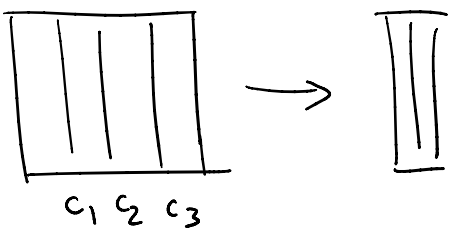


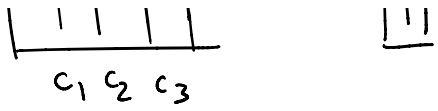
Selection (where)



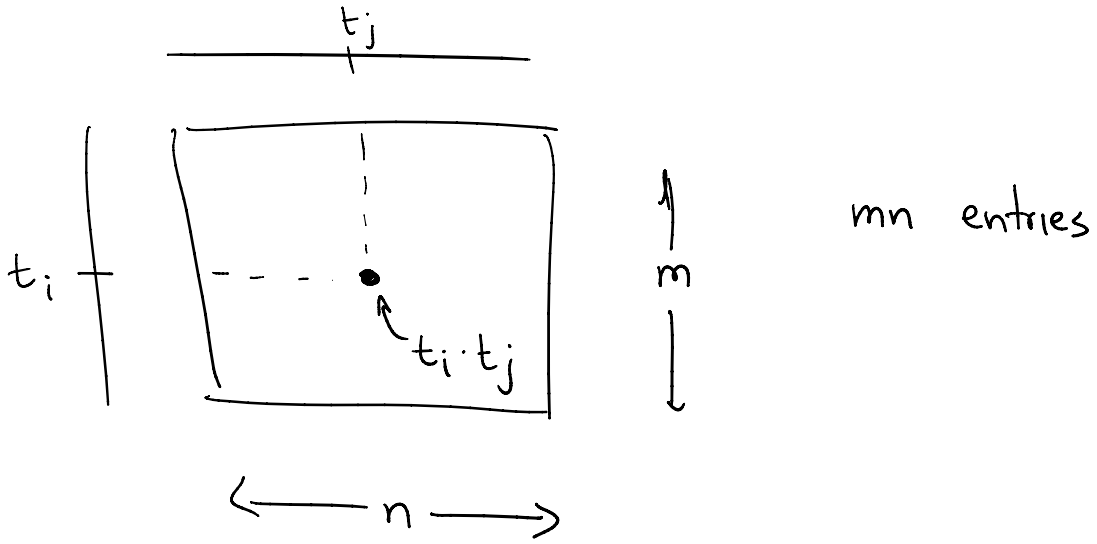
pred

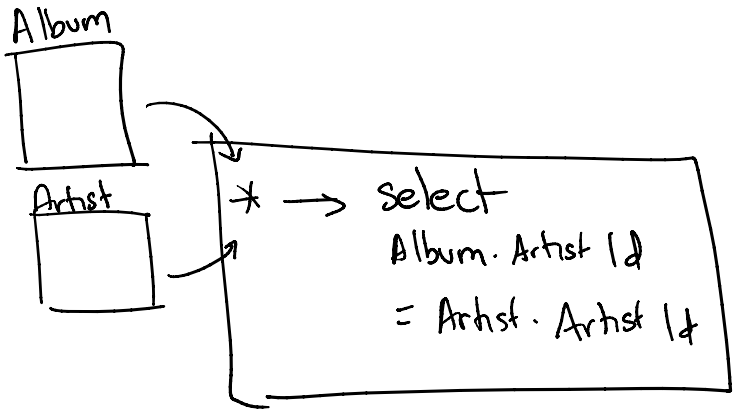
Projection (Select)

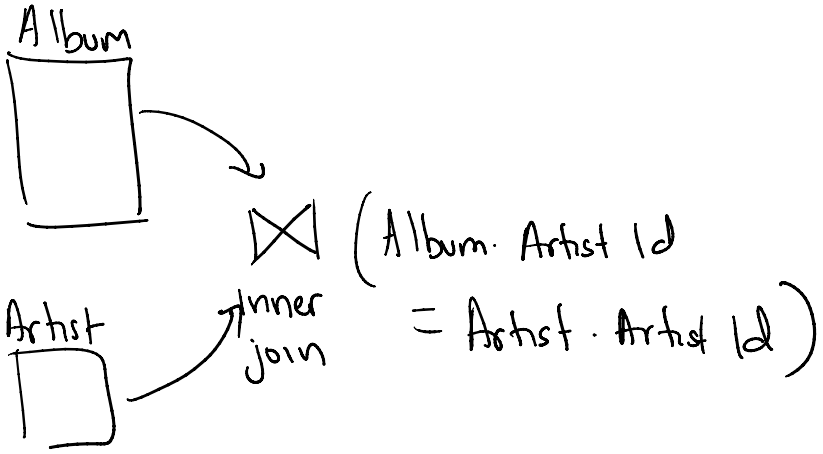




Cartesian product







DA Select Project Joins Union Intersect

RA	Select	Project	Joins, Cartesian product	Union	Intersection
SQL	where	select	Join(s)	Union	Intersect where exists
	σ	π	$\bowtie$ $*$ $\overline{\bowtie}$	$\cup$	$\cap$

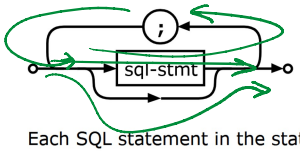
difference

minus

\

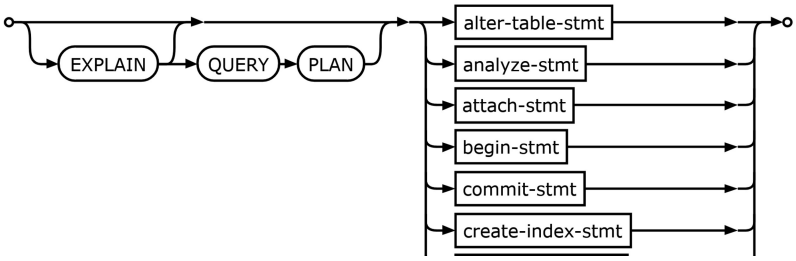
Railroad Diagram of SQLite's SQL dialect

sql-stmt-list:



Each SQL statement in the statement list is an instance of the following:

sql-stmt:



sql-stmt-list ::= empty str
| sql-stmt
| ; sql-stmt-list
| sql-stmt ; sql-stmt-list

Defining Relational Algebra

Table = set of tuples

$Q ::= T$ (Just a table)

$| \quad Q_1 \cup Q_2$ (Union)

$| \quad Q_1 \cap Q_2$ (Intersection)

$| \quad Q_1 \setminus Q_2$ (Set difference)

$| \quad \sigma_{filter}\ Q$ (Select)

$| \quad \Pi_{cols}\ Q$ (Project)

$| \quad Q_1 * Q_2$ (Cartesian product)

$| \quad Q_1 \bowtie Q_2$ (join)

- Inner
- outer
- left outer
- natural

Ex: All employees celebrating their birthday today

$$\sigma_{dob = today}\ Employee$$

Ex: Funding a travel subsidy

$$\sigma_{\text{salary} < 90K \wedge \text{daily commute} > 40\text{min}} \quad \text{Employee}$$

Ex: Interstate commuters

Employee

Name	Home Zip	Work Zip

Zip

Zip	State

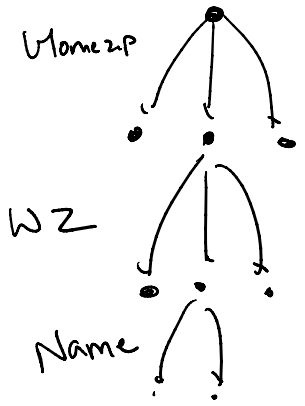
$$\sigma_{Z_1.S \neq Z_2.S}\left(\Pi_{E.N,\, Z_1.S,\, Z_2.S}\left(\left(E \bowtie_{E.H2 = Z_1.Z} Z_1\right) \bowtie_{E.W2 = Z_2.Z} Z_2\right)\right)$$

Evaluating RA Queries (at 20000ft)

Evaluating RA Queries (at 20000ft)

Employee

Name	Homezip	Workzip



$Q_1 \cup Q_2$: result := Q_1

for $t \in Q_2$

if $t \notin$ result :

result := result $\cup \{t\}$

$Q_1 \cap Q_2$: result := $\emptyset$

for $t_1 \in Q_1$:

if $t_1 \in Q_2$:

result := result $\cup \{t_1\}$

σ_f Q : result := $\emptyset$

for t in Q:

if f t:

result := result $\cup \{t\}$

Π_{cs} Q : ...

$Q_1 \bowtie_{Q_1.c_1 = Q_2.c_2} Q_2$: for t_1 in Q_1:

for all matchable t_2 in Q_2:

emit $(t_1 \cdot t_2)$

Loop Nest Join Accelerated using indices

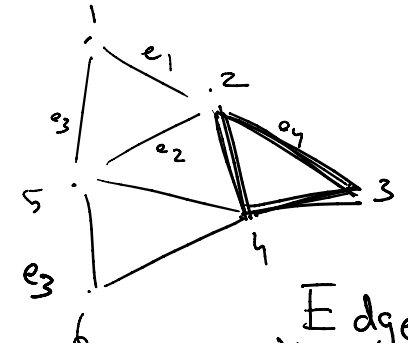
Hash join

Question: How do we evaluate joins quickly?

Triangle Query

select n_1 n_2 n_3

from edge e_1 edge e_2 edge e_3



Edge

from edge e_1 edge e_2 edge e_3

where $e_1 \cdot v_1 = e_2 \cdot v_2$ and

$e_2 \cdot v_1 = e_3 \cdot v_2$ and

$e_3 \cdot v_1 = e_1 \cdot v_2$

$n_1 = e_1 \cdot v_1 \quad n_2 = e_2 \cdot v_1 \quad n_3 = e_3 \cdot v_1$

Edge v_1	v_2
1	5
1	2
5	2
2	3
4	6
⋮	

---

for $e_1 \in$ Edge

for $e_2 \in$ Edge

if $e_1 \cdot v_2 = e_2 \cdot v_1$ and

if $(e_2 \cdot v_2, e_1 \cdot v_1) \in$ Edge,

then emit $(e_1 \cdot v_1, e_2 \cdot v_1, e_2 \cdot v_2)$

Triangle Query :

- There is an algorithm which takes $O(n^2)$ time

- There exist Edge tables where result contains $O(n^{1.76})$ tuples

result contains $O(n^{1.76})$ tuples