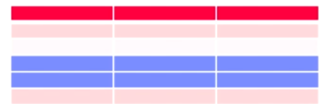
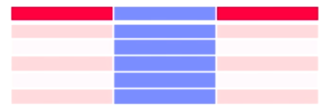
Partitioning Strategies

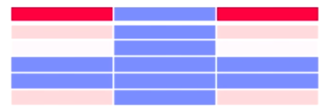
- **Horizontal Partitioning**
  - Relation partitioning into disjoint subsets



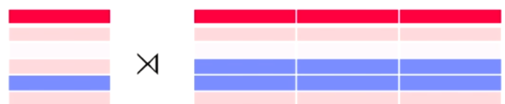
- **Vertical Partitioning**
  - Partitioning of attributes with similar access pattern



- **Hybrid Partitioning**
  - Combination of horizontal and vertical fragmentation (hierarchical partitioning)



- **Derived Horizontal Partitioning**



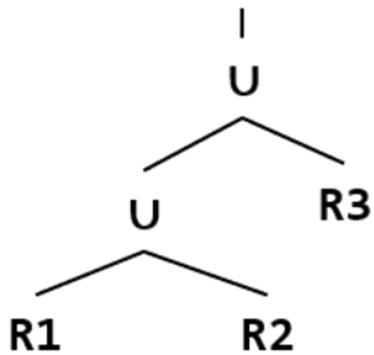
Correctness Properties

- relation R partitioned into n fragments
- completeness
  - each item from R must be included in at least one fragment
- reconstruction
  - exact reconstruction of fragments must be possible
- disjointness
  - no item must be in more than one fragment
  - $R_i \cap R_j = \emptyset$

Horizontal Partitioning

- row partitioning into n fragments
  - complete, disjoint and reconstructable
  - schema of fragments is equivalent to schema of base relation
- partitioning
  - split table by n selection predicates on attributes
    - e.g. split by last name
  - beware of attribute domain and skew

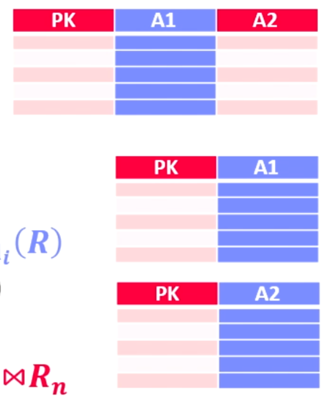
- reconstruction
  - union of all fragments
  - bag semantics but no duplicates across partitions



$$R = \bigcup_{1 \le i \le n} R_i$$

Vertical Fragmentation

- **Column Partitioning into n Fragments Ri**
  - **Complete**, **reconstructable**, but not disjoint (**primary key** for reconstruction via join)
  - Completeness: each attribute must be included in at least one fragment



- **Partitioning**
  - Partitioning via **projection**
  - Redundancy of primary key

$$R_i = \pi_{PK,A_i}(R) \quad (1 \le i \le n)$$

- **Reconstruction**
  - **Natural join** over primary key

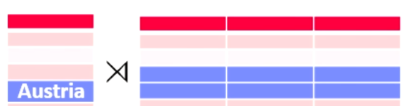
$$R = R_1 \bowtie R_i \bowtie R_n \quad (1 \le i \le n)$$

- **Hybrid horizontal/vertical partitioning**

$R = R_1 \bowtie R_i \bowtie R_n$ w/ $R_i = \cup R_{ij}$
→ $R = \cup R_j$ w/ $R_j = R_{1j} \bowtie R_{ij} \bowtie R_{nj}$

Derived Horizontal Fragmentation

- **Row Partitioning R into n fragements R_i, with partitioning predicate on S**
  - Potentially complete (not guaranteed), **restructable**, **disjoint**
  - Foreign key / primary key relationship determines correctness



- **Partitioning**
  - **Selection** on independent relation S
  - **Semi-join** with dependent relation R to select partition R_i

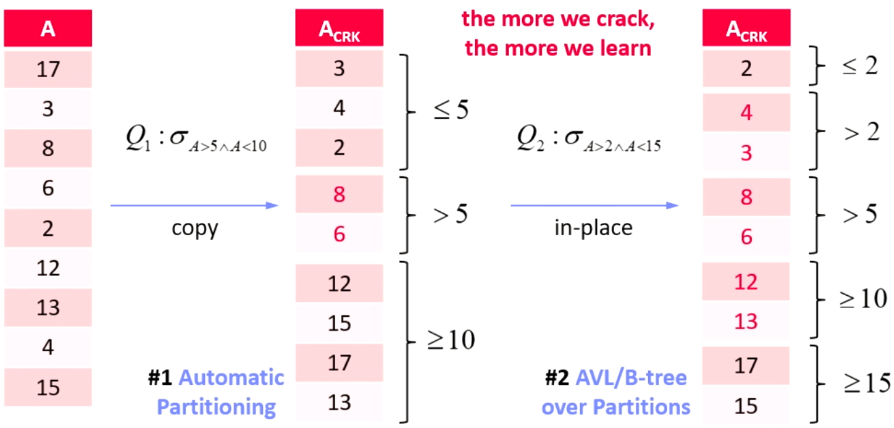
$$R_i = R \ltimes S_i = R \ltimes \sigma_{P_i}(S) = \pi_{R.*}\left(R \bowtie \sigma_{P_i}(S)\right)$$

- **Reconstruction**
  - Equivalent to horizontal partitioning
  - **Union** of all fragments

$$R = \bigcup_{1 \le i \le n} R_i$$

Exploiting Partitioning in Postgre

Database Cracking

- database layout adapts to requested queries and their range predicates
  - creates [[Index Structures]] to identify qualifying partitions
  - inside partition its elements are unordered
- workload creates hybrid between partitioning and indexing
- 

[[Background Storage System]] [[Database Performance Tuning]]