DATA 514 Section 6 Worksheet - Jonathan Jacobs

Relational Algebra

Question 1

- **σ (Selection)**: This operator filters rows in a relation based on a specific condition. It is used to select tuples that satisfy the given condition, effectively narrowing down the dataset.
- **⋈ (Join)**: This operator combines tuples from two different relations based on a condition that relates their columns. It is primarily used to perform equi-joins and natural joins, facilitating the merging of related data from separate sources.
- **δ (Duplicate Removal)**: This operator removes duplicate rows from a relation to ensure that each tuple in the output is unique. It is particularly useful in scenarios where redundancy can lead to inaccuracies or inefficiencies.
- **π (Projection)**: This operator selects certain columns from a relation, thereby reducing the number of columns in the output. It can increase the number of duplicate rows if the selected columns do not uniquely identify tuples.
- **γ (Grouping and Aggregation)**: This operator groups rows that share the same values in specified columns and performs aggregations like sum, count, max, and min on other columns. It is essential for summarizing and analyzing data at a grouped level.

Question 2

1. **FROM and WHERE (Join)**:
   - **Expression**: (R ⋈_{R.b = T.b} T)
   - **Description**: Join Table_R (R) and Table_T (T) on the condition R.b equals T.b. This combines records from both tables where the join condition is met.
2. **GROUP BY and SELECT (Projection and Grouping)**:
   - **Expression**: (γ_{R.b, T.c; \text{max}(T.a) \rightarrow T_{max}} (R ⋈_{R.b = T.b} T))
   - **Description**: Group the result of the join by columns R.b and T.c and calculate the maximum of T.a, labeling it as T_max. This step aggregates data by specified keys and computes aggregated metrics.
3. **HAVING (Selection)**:
   - **Expression**: (σ_{T_{max} > 99} (γ_{R.b, T.c; \text{max}(T.a) \rightarrow T_{max}} (R ⋈_{R.b = T.b} T)))
   - **Description**: Apply a filter to keep only those groups where T_max is greater than 99. This selection criterion further refines the dataset to meet specific conditions.

Question 3

1. **FROM and WHERE (Selection)**:
   - **Expression**: (\sigma_{\text{fname} = 'Patrick' \land \text{lname} = 'Stewart'} (Actor))
   - **Description**: Apply a selection operator to the `Actor` table to filter rows where the first name is 'Patrick' and the last name is 'Stewart'.
2. **SELECT (Projection)**:
   - **Expression**: (\pi_{\text{fname, lname, age}} (\sigma_{\text{fname} = 'Patrick' \land \text{lname} = 'Stewart'} (Actor)))
   - **Description**: Project the columns `fname`, `lname`, and `age` from the rows filtered in the previous step. This results in a relation that contains only the specified attributes of actors named Patrick Stewart.

Question 4

1. **FROM and WHERE (Joins and Selection)**:
   - **Expression**: (\sigma_{A.\text{age} < 30} (Actor \times ActsIn \times Movie))
   - **Description**: Cartesian product of `Actor`, `ActsIn`, and `Movie`, followed by selection where `Actor.aid` matches `ActsIn.aid`, `Movie.mid` matches `ActsIn.mid`, and `Actor.age` is less than 30.
2. **GROUP BY and SELECT (Projection, Grouping, and Aggregation)**:
   - **Expression**: (\gamma_{M.\text{mid}, M.\text{name}; \text{COUNT}(*) \rightarrow \text{cnt}} (\sigma_{A.\text{age} < 30 \land A.\text{aid} = AI.\text{aid} \land M.\text{mid} = AI.\text{mid}} (Actor \times ActsIn \times Movie)))
   - **Description**: Group the result of the join by `Movie.mid` and `Movie.name`, and calculate the count of records for each group. Project the movie name and the count as cnt.
3. **HAVING (Selection)**:
   - **Expression**: (\sigma_{\text{cnt} > 1} (\gamma_{M.\text{mid}, M.\text{name}; \text{COUNT}(*) \rightarrow \text{cnt}} (\sigma_{A.\text{age} < 30 \land A.\text{aid} = AI.\text{aid} \land M.\text{mid} = AI.\text{mid}} (Actor \times ActsIn \times Movie))))
   - **Description**: Apply a selection to filter groups where the count is greater than 1, thus only returning movies with more than one young actor (under 30 years old) involved.

Cardinality Estimation

Question 5

Optimized RA Tree

1. **Pre-filter `Users` and `Packages`**:
   - **Users Filter**: (\sigma_{\text{UserID} = 'Amal'} (Users)) - Isolates data for 'Amal', significantly reducing the number of rows processed from the `Users` table.
   - **Packages Filter**: (\sigma_{\text{NumItems} = 7} (Packages)) - Narrows down the `Packages` table to entries with `NumItems` = 7 before the join.

2. **Efficient Join**:
   - **Expression**: ((\sigma_{\text{UserID} = 'Amal'} (Users)) \⋈_{UserID = UserID} (\sigma_{\text{NumItems} = 7} (Packages)))
   - **Description**: Join the filtered subsets, reducing the computational load by combining only the necessary data.
3. **Projection**:
   - **Expression**: (\pi_{UserID, PackageID} (\text{Join Result}))
   - **Description**: Projects the relevant `UserID` and `PackageID`, finalizing the query's output.

This approach prioritizes early filtering to manage the vast data size effectively, ensuring that only essential data enters the join operation, which optimizes performance significantly.

Question 5

Optimized RA Tree

1. **Pre-filter `Singers`**:
   - **Expression**: (\sigma_{\text{name} = 'Taylor Swift'} (\text{Singers}))
   - **Description**: Immediately filter the `Singers` table for 'Taylor Swift', which significantly reduces the dataset size before joining.
2. **Efficient Join**:
   - **Expression**: ((\sigma_{\text{name} = 'Taylor Swift'} (\text{Singers})) \⋈_{\text{sid} = \text{sid}} \text{Albums})
   - **Description**: Join the filtered `Singers` with `Albums` on `sid`, optimizing the join operation by reducing the amount of data processed.
3. **Projection**:
   - **Expression**: (\pi_{\text{title}} (\text{Join Result}))
   - **Description**: Project the `title` of albums from the join result, displaying only albums by 'Taylor Swift'.

This approach minimizes processing by filtering early and efficiently joining smaller datasets, enhancing query performance.