Table API

Felix Neutatz
neutatz@gmail.com

Slides by Aljoscha Krettek

SQL rocks!

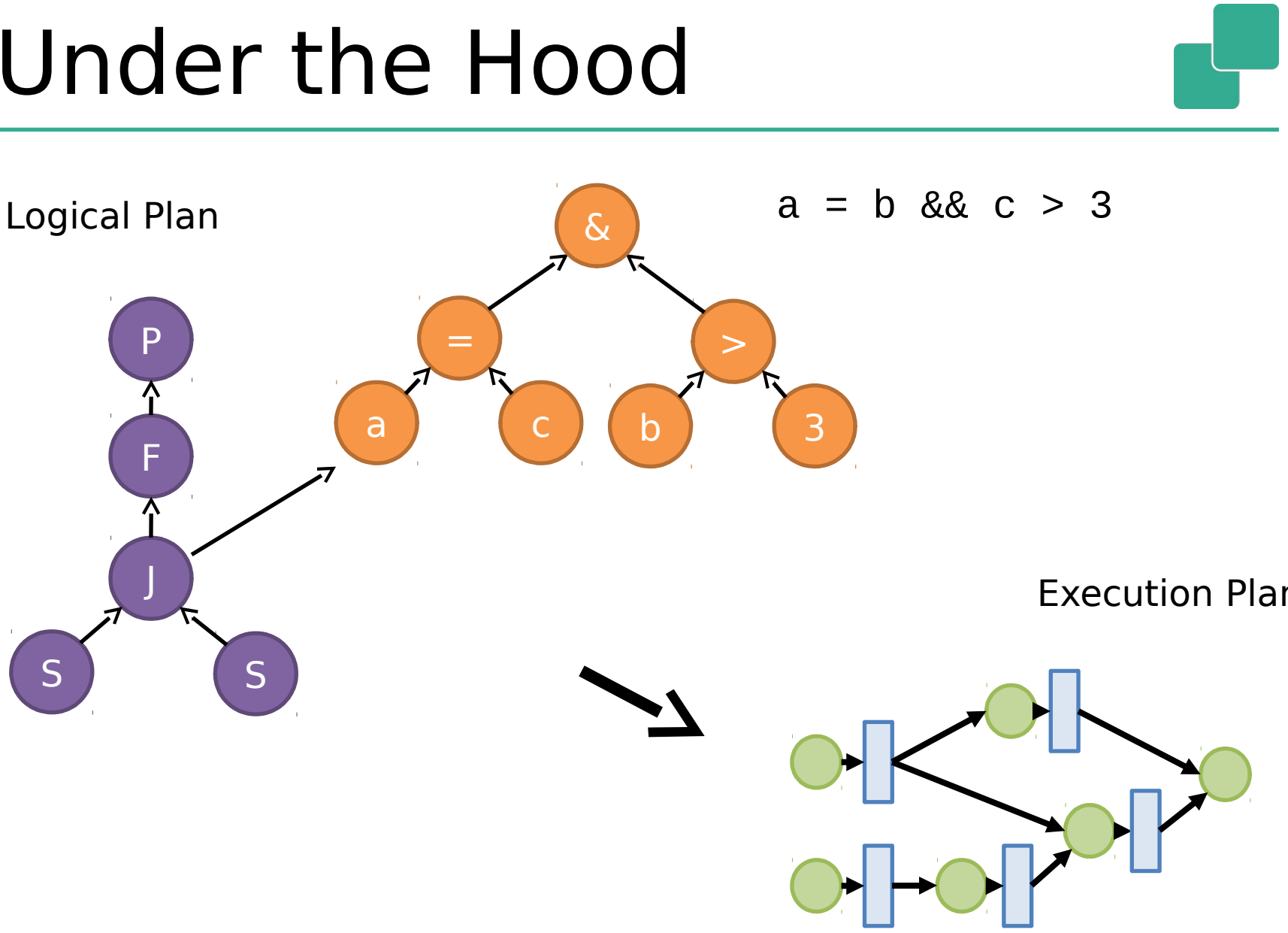
- “Everybody” knows SQL
- All major data management and database systems support SQL

→ TableAPI

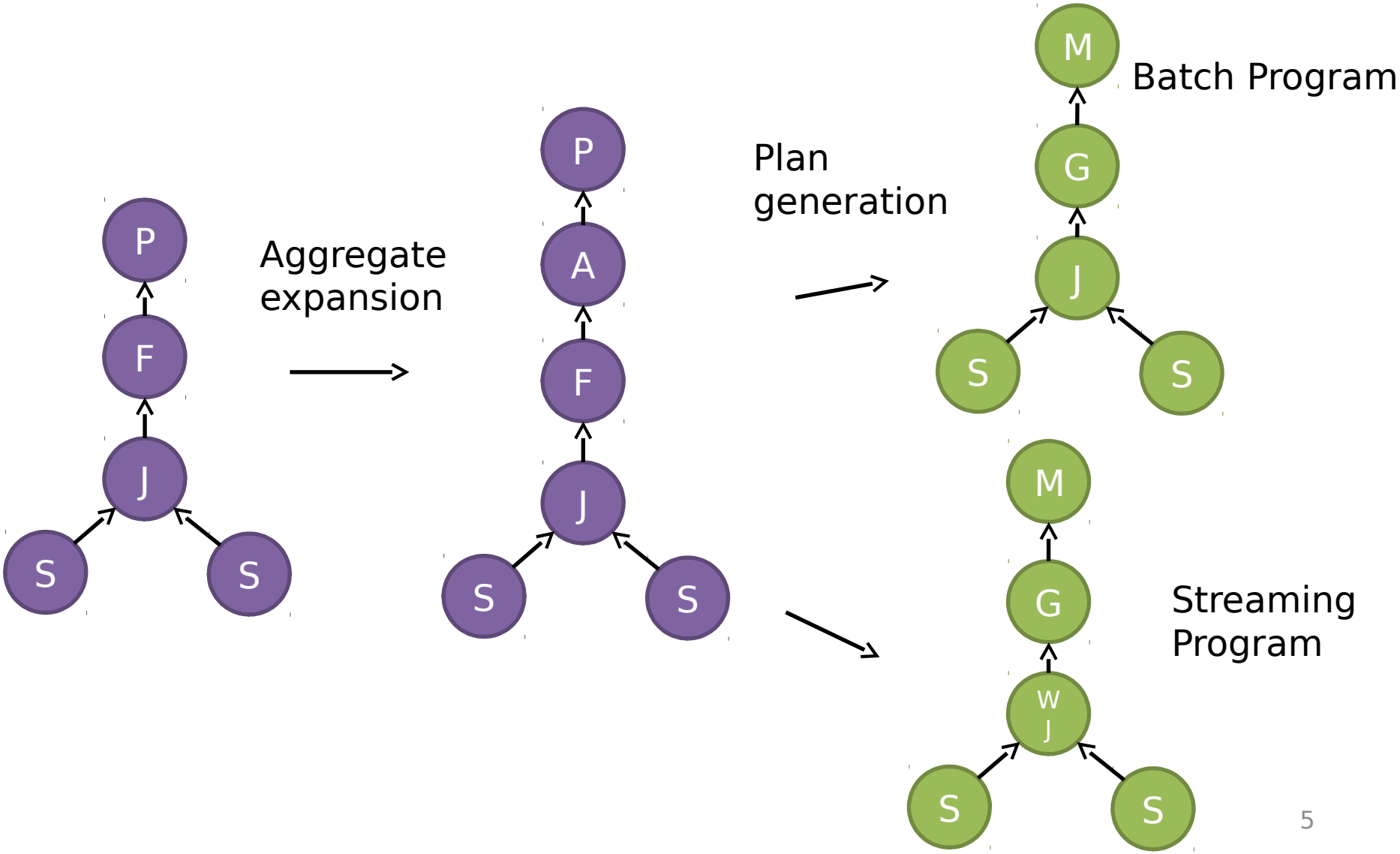
First Things First

```
Table clicksTable = tableEnv.toTable(clicks, "url, userId");

Table clickCounts= clicks
  .groupBy("user")
  .select("userId, url.count as count")
  .filter("count > 4");


Table activeUsers = users.join(clickCounts)
  .where("id = userId && count > 10")
  .select("username, count");
```

Under the Hood



Plan Translation



String operations

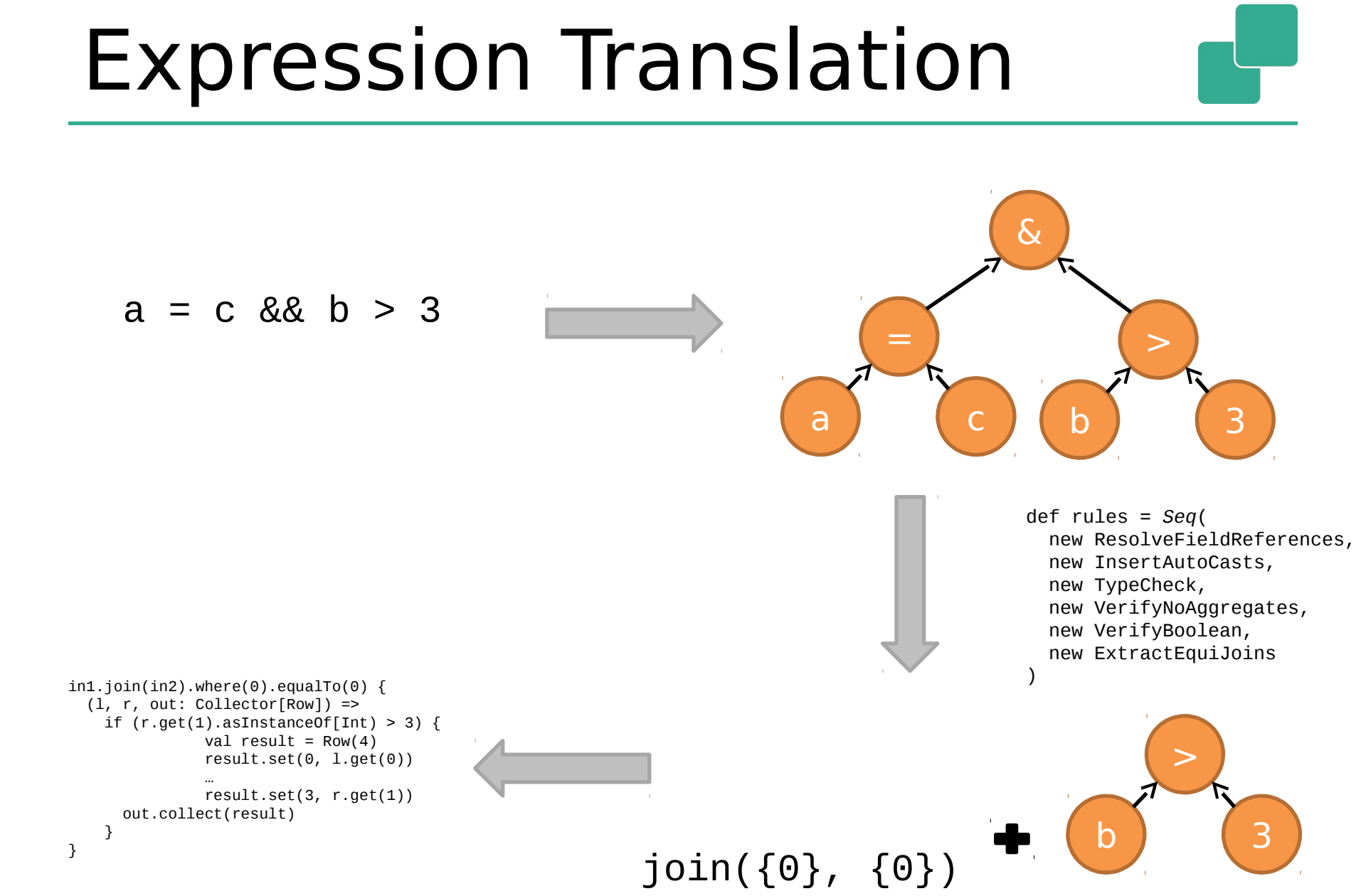
- Sub strings: "hello world" → "hello"

```
in.select("a.substring(0, b.avg + 3) as aShort")
```

- String concatination: "hello" + " world" → "hello world"

```
in.select("a.count + ' is the count'")
```

Expression Translation



Back and Forth

```
public static class Out {
       public String c;
       public Integer d;
}
…

DataSet<Tuple2<Integer,String>> input = …

TableEnvironment tableEnv = new TableEnvironment();
Table in = tableEnv.toTable(input,"a,b");
Table result = in.groupBy("b").select("b, a.avg”);

DataSet<Out> result =
       tableEnv.toSet(result.as("c,d"), Out.class);
```

- Supports POJOs, Case classes, Tuples

What Works?

- Relational queries from both Java and Scala
- Translation to batch programs
- Preliminary translation to streaming programs

Future Work

- Relational Optimization
  - Filter/Projection push down
  - Join order
- Operator Fusion
- Extend expressions
  - string operations, casting, explode/gather, date/time, ...
- Windowing operations (streaming)
- Columnar execution?
- SQL

Hands-on

- Task:
- A German bus company asks for your help.
- They provide 20 movies in their bus entertainment center. They want to have a nice chart about the average rating for all movies which have been rated by more than 1000 passengers.

- Download the git repository:

  git clone https://github.com/FelixNeutatz/flink-hands-on-TableAPI.git

- Start IntelliJ → File → Open → ../flink-hands-on-TableAPI/pom.xml
- Complete de.tuberlin.dima.flinkhandson.tables.TablesSolution

- Bonus Task: Generate a similar chart for a single genre :)