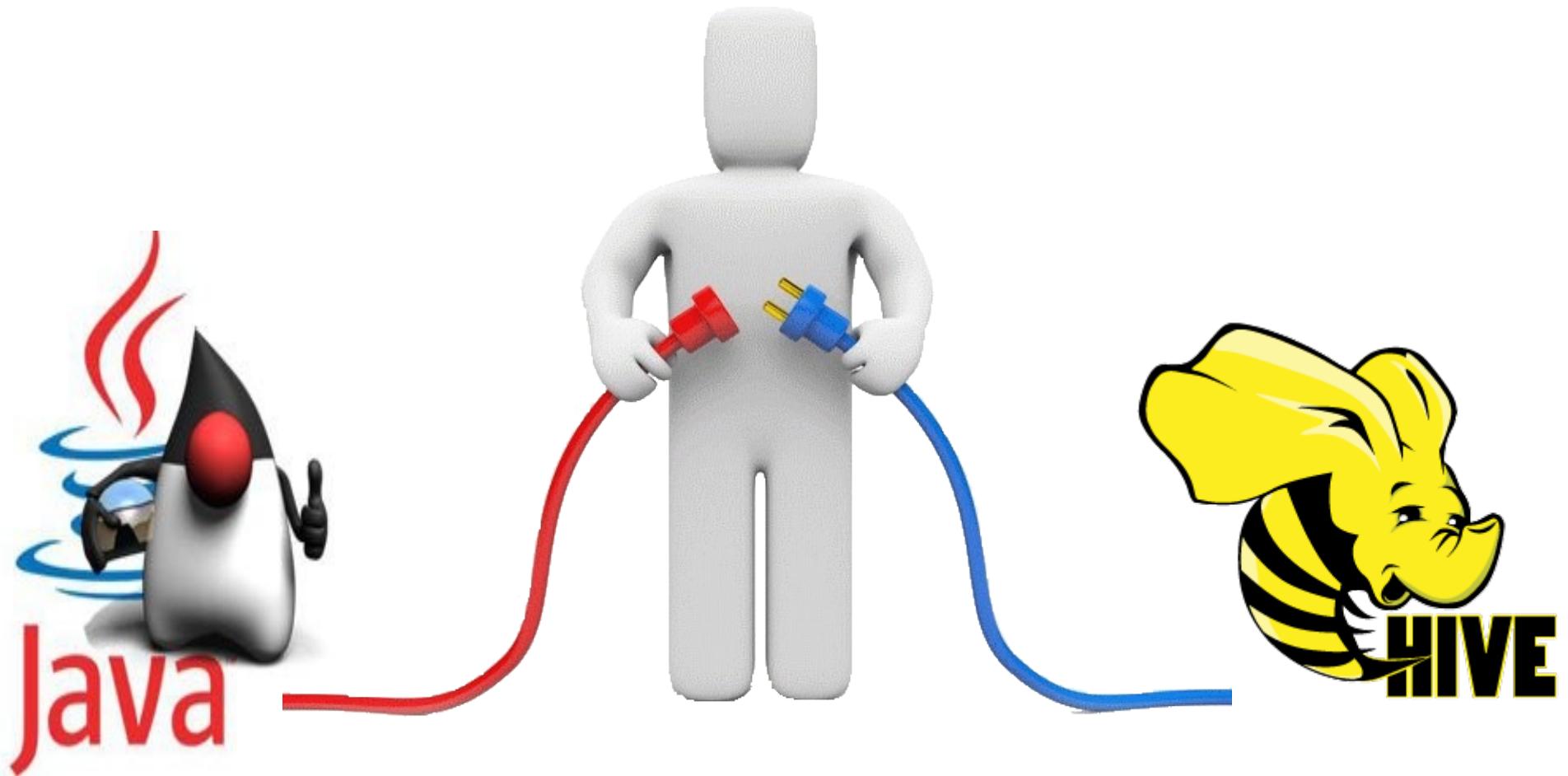


# Using jdbc with hive



# Agenda

- Who are we
- Jdbc for ETL
- Jdbc for BI people
- Questions

# eBuddy



- Web based chat
  - Started in 2003 (no statistics)
  - 1.7B record (im logins)
    - Started basic logging in 2004
- XMS
  - 490M record (xms)
    - Launched May 23, 2011
- Interesting to know
  - Hosting in the US but BI people in Amsterdam
  - Developers are Java centric

# ETL

- Connection pooling
  - InitSql (new connections with udfs)
- Jdbc templates (Spring)

# ETL – Connection pooling

```
<bean id="dwhHiveDataSource" class="org.apache.commons.dbcp.BasicDataSource" destroy-method="close">
    <property name="driverClassName" value="${db.dwh.class}"/>
    <property name="url" value="${db.dwh.url}"/>
    <property name="username" value="${db.dwh.user}"/>
    <property name="password" value="${db.dwh.pwd}"/>
    <property name="minIdle" value="${db.dwh.min}"/>
    <property name="testOnBorrow" value="${db.dwh.testonborrow}"/>
    <property name="validationQuery" value="${db.dwh.validationquery}"/>
    <property name="connectionInitSqls" ref="listInitSql"/>
</bean>

<util:list id="listInitSqlLocal">
    <value>SET pool.name=${pool.name}</value>
    <value>SET hive.exec.mode.local.auto=${hive.exec.mode.local.auto}</value>
    <value>add jar ${udf.location}</value>
    <value>create temporary function iptolong as 'com.ebuddy.dwhhive.udf.IpToLong'</value>
</util:list>
```

# ETL - Connection pooling

```
public class ImEtlServiceImpl extends JdbcDaoSupport
```

Then wire it in:

```
<bean id="im"
class="com.ebuddy.dwhhive.etl.im.ImEtlServiceImpl">
  <property name="dataSource" ref="dwhHiveDataSource"/>
</bean>
```

# ETL - Jdbc templates

```
String sql = "drop table sometable";  
getJdbcTemplate().execute(sql);
```

```
String sql = "select count(*) from sometable";  
long recordCnt = getJdbcTemplate().queryForLong(sql);
```

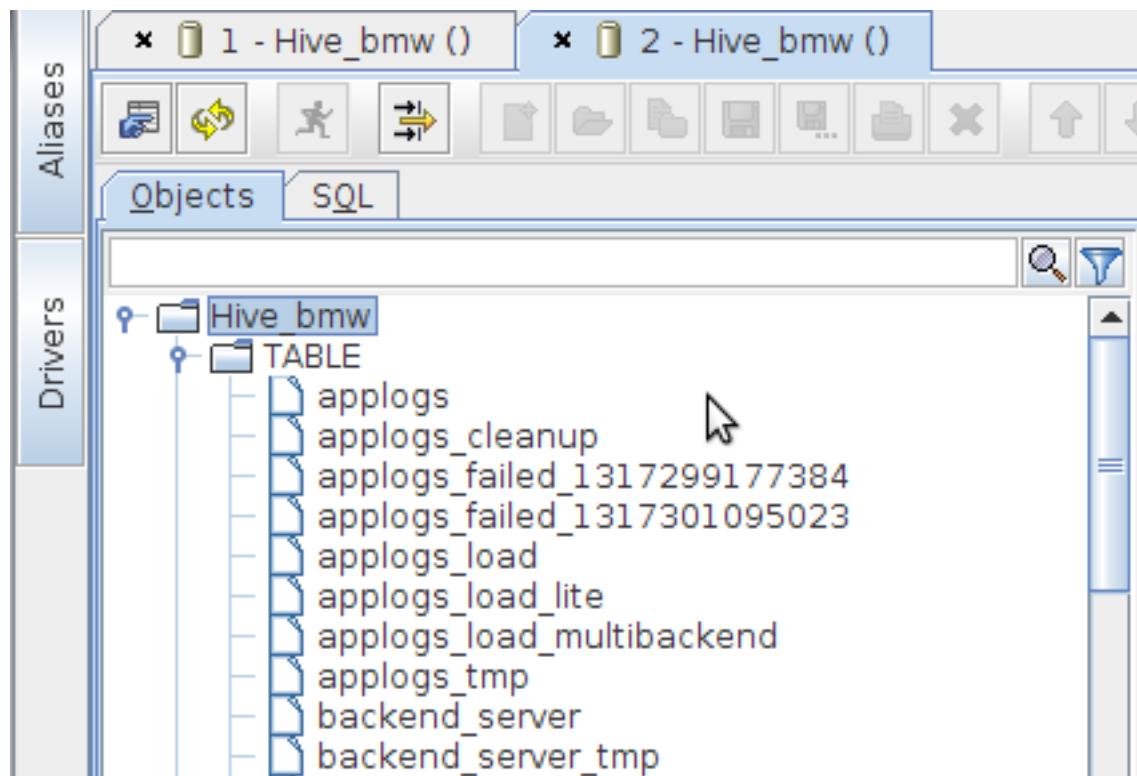
```
String sql = "select * from sometable";  
List<Map<String, Object>> recordsLast =  
getJdbcTemplate().queryForList(sql);
```

# Squirell

- Meta data (HIVE-1126)
  - List of existing tables/data types
  - Code completion (Ctrl-Space)
- Concurrency
- Performance
  - “Building Output” (HIVE-1815)



# Squirell - getTables



# Squirell – View data

The screenshot shows the Squirell SQL IDE interface. At the top, there are two tabs labeled "1 - Hive\_bmw ()" and "2 - Hive\_bmw ()". Below the tabs is a toolbar with various icons for database management tasks. The main area is divided into two sections: "Objects" on the left and "SQL" on the right. The "Objects" section lists several tables: client\_tmp, country, country\_old, groupsessions, logerror, logintype, logline, and logtype. The "country" table is currently selected, indicated by a blue border around its name. The "SQL" section contains a data viewer for the "country" table. The viewer has a header row with columns: Exported Keys, Imported Keys, Indexes, Privileges, Column Privileges, Row IDs, and Version. Below this is a sub-header with tabs: Info, Content, and Row Count. The "Content" tab is active, showing the following data:

country_id	code	name	region	population
0	--	Unknown	Unknown	<null>
1	SE	Sweden	Western Europe	9031088
2	IT	Italy	Western Europe	58147733
3	A2	Satellite Provider	Unknown	<null>
4	BD	Bangladesh	Asia	150448339

# Squirell – code completion

The screenshot shows the Squirell SQL Editor interface. On the left, there are tabs for 'Aliases' and 'Drivers'. The main area has two tabs at the top: 'Objects' (selected) and 'SQL'. Below them is a code editor containing the following SQL:

```
select * from xms_tmp_delta_1319708465793
select * from cou
```

The word 'cou' is highlighted in red, indicating it is a misspelling or a placeholder. A dropdown menu is open over the word 'cou', listing three suggestions:

- country\_id int(10) NULL
- country
- country\_old

# Squirell - “Building Output”

Records	Fetch size	Time on output
1000	1	2min 43sec
1000	50	4sec
1000	100	2sec
1000	1000	1sec

- Suboptimal

# QR code

presentation



xms

