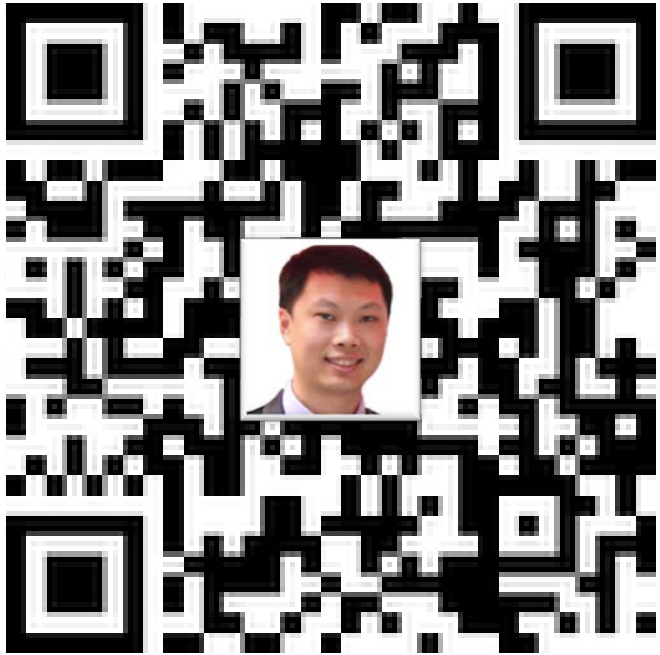
NoSQL、RDS和大数据异构融合实例分享

关于我

萧少聪(铁庵) Scott Siu
中国广东 · 中山人

Postgres中国用户会 创始人之一
阿里云ApsaraDB PG/PPAS产品经理

EnterpriseDB认证数据库专家
RedHat RHCA认证架构师



关注微博
(@萧少聪Postgres)

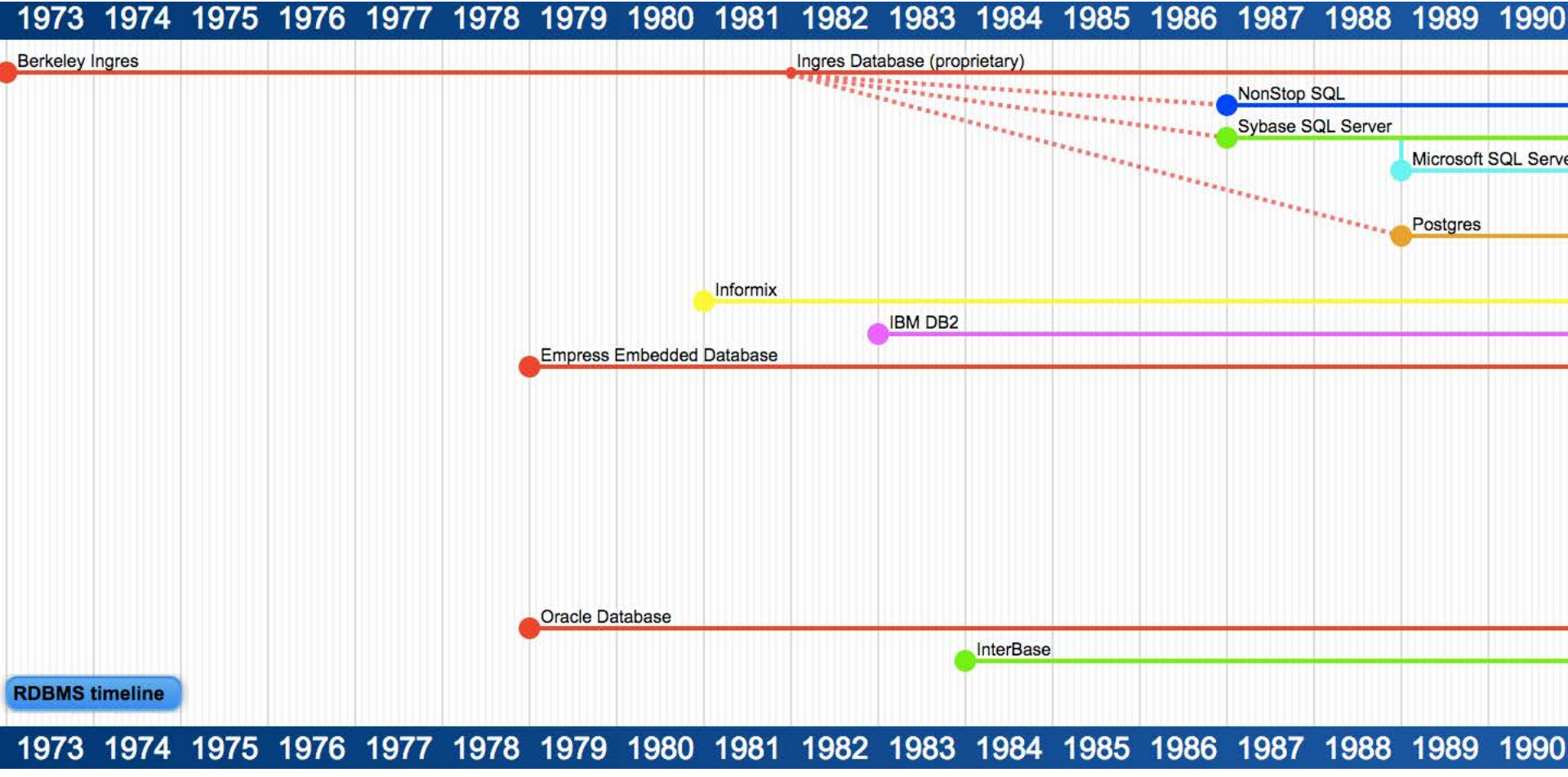
阿里云 aliyun.com

PostgreSQL

提　纲

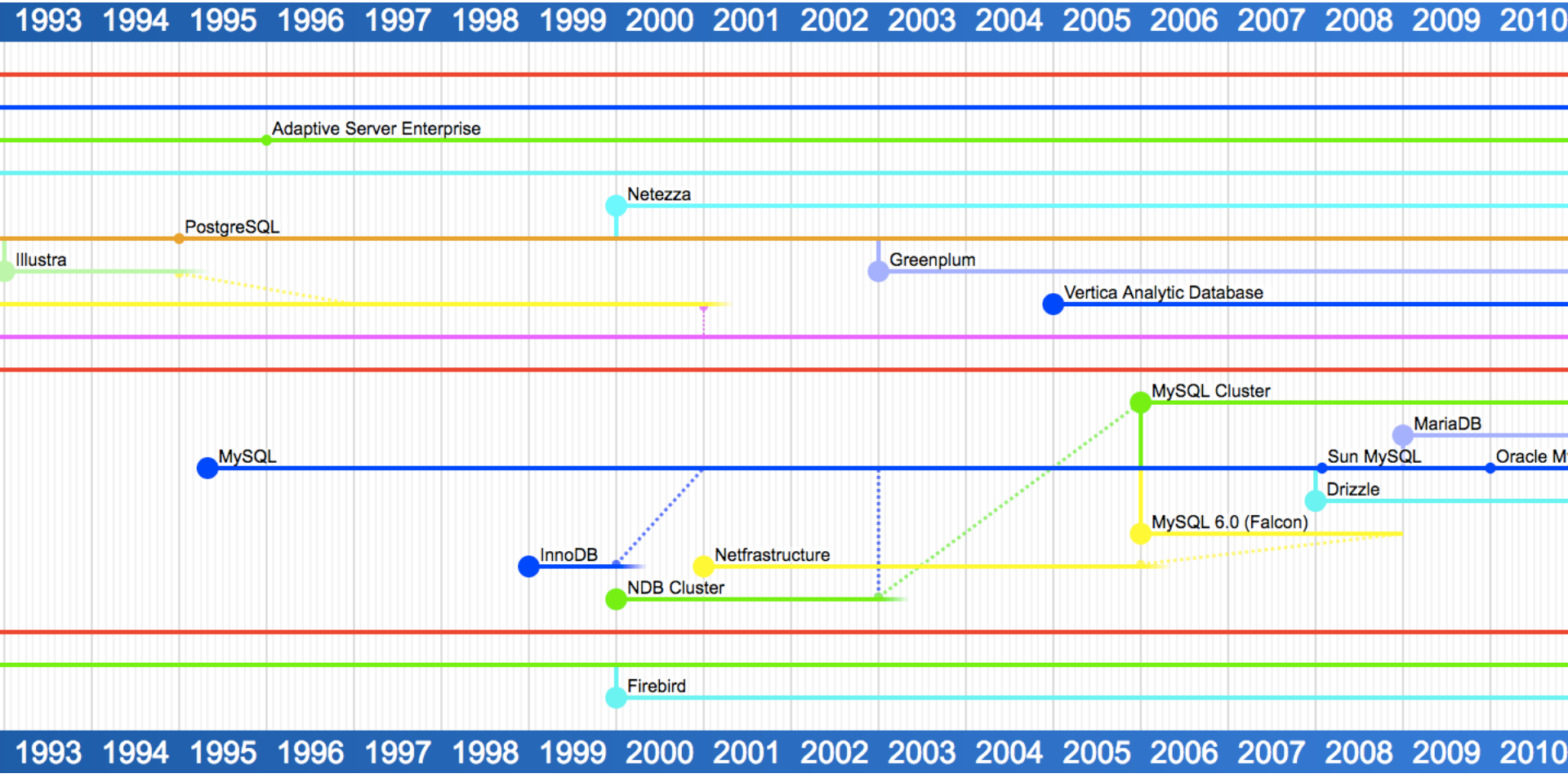
- Why Postgres
- Why FDW
- FDW: SQL Everything加速传统行业转型
- FDW: 金融报文处理
- FDW: 物联网数据整合
- FDW: 解决企业并购重组问题

Why Postgres - History



参考：http://en.wikipedia.org/wiki/File:RDBMS_timeline-2.svg

Why Postgres - History



参考：http://en.wikipedia.org/wiki/File:RDBMS_timeline-2.svg

Why Postgres - 开放

- 基于BSD/MIT协议：对所有人免费，可以任意处置，包括使用，复制，修改，合并，发表，分发，再授权，或者销售。唯一的限制是，软件中必须包含上述版权和许可提示。
- 比其它开源协议如 GPL / Apache 更开放适合于长远的企业战略发展

Why Postgres - 稳定

- 多程衍生产品已经用于传统金融、通讯、能源企业认可，如：

  平安集团(平安科技)、MasterCard

  浙江移动、日本NTT、韩国KT

  国家电网

- 超过40年历史，专注于OLTP事务处理

Why Postgres - 企业功能

- 数据流式复制（9.6支持多节点全同步）
- 基于ACID的JSON及KV数据类型
- PostGIS地理信息模型
- FDW对接第三方数据源，或实现Sharding
- 可扩展支持R语言、网络、生物、化学等函数
- 唯一明确说明对SQL 2011的兼容程度>90%

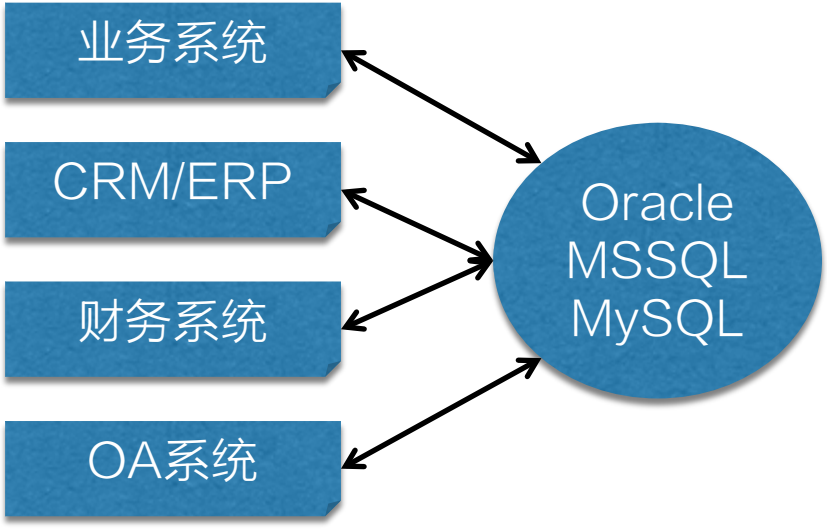
Why Postgres - 机遇

- 中国使用量相比国外为 1:30 潜力巨大
- 中国查询指数比2011年提高5倍以上
- 众多国内外商业数据库都基于 Postgres，生态丰富，大有成为数据库界 Linux 之势
- 国家自主、可控、开源的风向支持

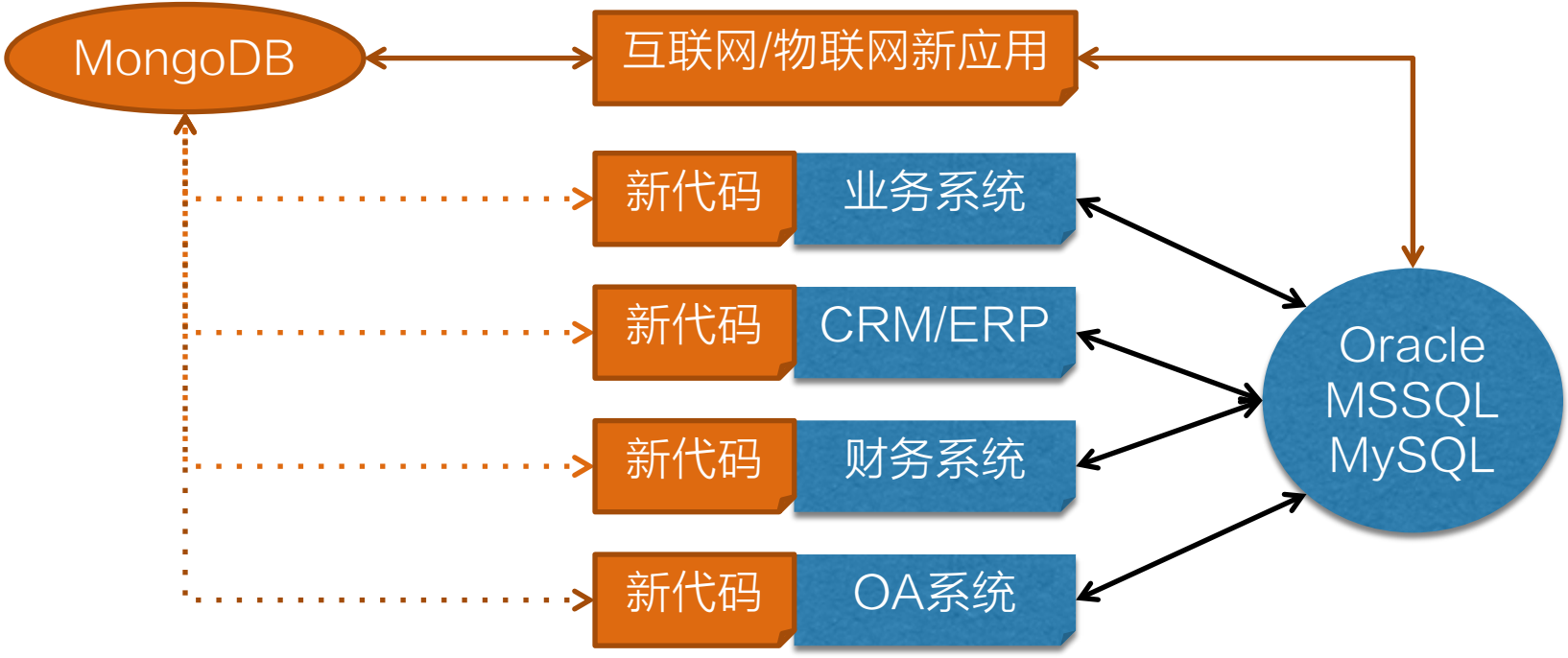
提　纲

- Why Postgres
- Why FDW
- FDW: SQL Everything加速传统行业转型
- FDW: 金融报文处理
- FDW: 物联网数据整合
- FDW: 解决企业并购重组问题

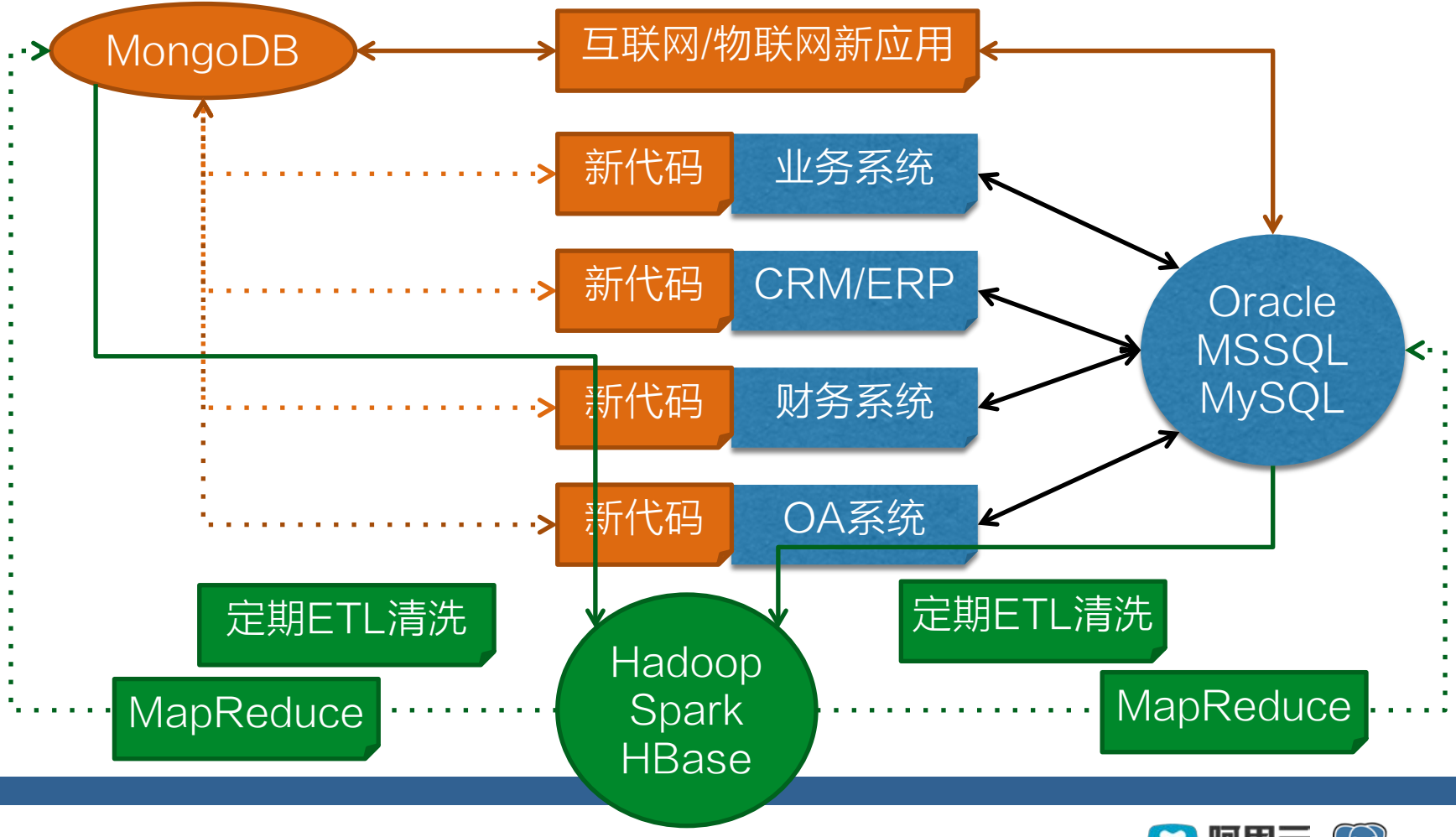
Why FDW - 传统企业现有的数据操作模型



Why FDW - 传统企业面向互联网/物联网需求



Why FDW - 传统企业整合大数据处理



Why FDW - 传统企业(ISV)面临的问题总结

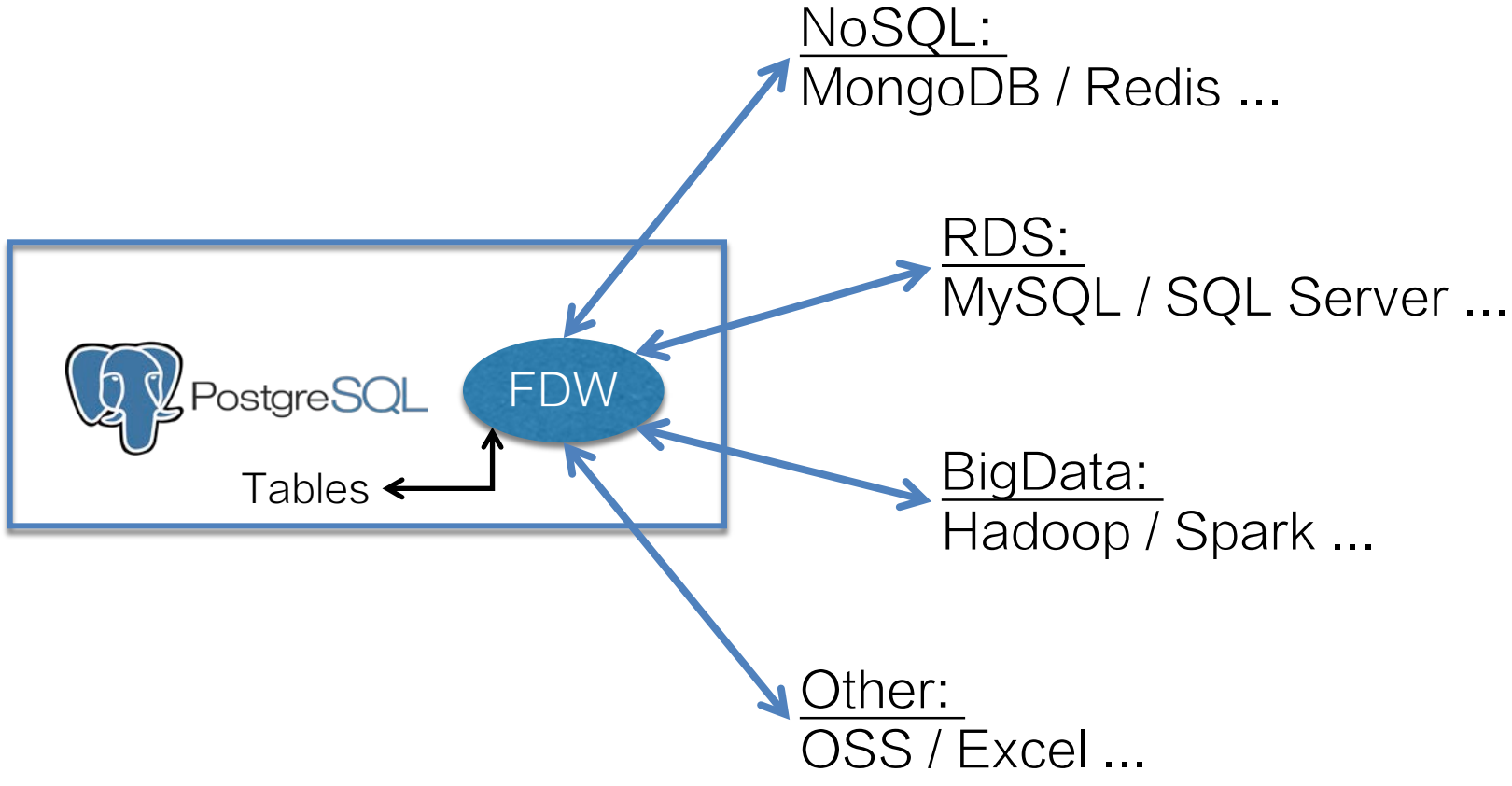
- 时间：不抓紧时间转型将可能错失良机
- 复杂：新架构跨界操作多管理难
- 团队：新知识积累不足，抗风险能力低

Why FDW - 什么是Postgres FDW

- FDW全称Foreign-Data Wrapper
- 简单来说就是PostgreSQL中的外部表功能
- 得益于PostgreSQL源代码的开放性，当前已经支持包括：MongoDB、Redis、MySQL、SQL Server、Oracle、Hadoop、Hive、Elastic Search等近30种不同的外部数据源

参考：https://wiki.postgresql.org/wiki/Foreign_data_wrappers

Why FDW - 什么是Postgres FDW



提　纲

- Why Postgres
- Why FDW
- FDW: SQL Everything加速传统行业转型
- FDW: 金融报文处理
- FDW: 物联网数据整合
- FDW: 解决企业并购重组问题

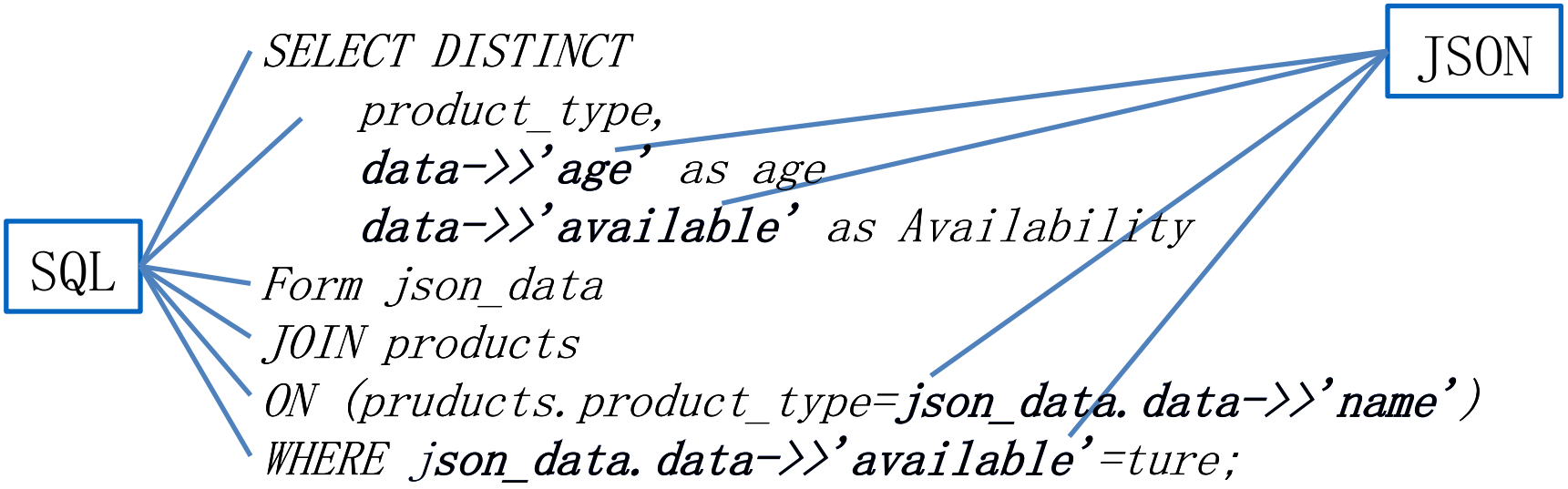
借助Postgres协助实现“敏捷项目”转型

- 转型不是一步到位，是持续调整
- 敏捷：以最小成本试错，快速迭代
- 打破NoSQL和BigData的障碍，快速转型
- 再进行后续优化，快速进入新领域
- 发挥现有团队有SQL价值：

  SQL Everything加速传统行业转型

SQL + NoSQL(JSON) = NewSQL

```
CREATE TABLE json_data (data jsonb);
INSERT INTO json_data VALUES ('{"name":"xxx","age":"28", "available":true}');
```



JSON的背景

JSON(JavaScript Object Notation) 是一种轻量级的数据交换格式。 易于人阅读和编写。同时也易于机器解析和生成。 它基于JavaScript Programming Language, Standard ECMA-262 3rd Edition - December 1999的一个子集。JSON采用完全独立于语言的文本格式，但是也使用了类似于C语言家族的习惯（包括C, C++, C#, Java, JavaScript, Perl, Python等）。 这些特性使JSON成为理想的数据交换语言。

JSON 已经是 JavaScript 标准的一部分。目前，主流的浏览器对 JSON 支持都非常完善。应用 JSON，我们可以从 ~~XML~~ 的解析中摆脱出来，对那些应用 Ajax 的 Web 2.0 网站来说，JSON 确实是目前最灵活的轻量级方案。

JSON的 背景

XML

```
<book>
        <type>textbook</type>
        <pages>256</pages>
        <title>Programming Pearls 2nd Edition</title>
        <description>The first edition of Programming Pearls
was one of the most influential books I read early in my
career...</description>
        <rating>4.5</rating>
        <coverType>paperback</coverType>
        <genre>Computer Science</genre>
        <author>Jon Bentley</author>
        <publisher>Addison-Wesley Professional</publisher>
        <copyright>1999</copyright>
</book>
```

JSON的 背景

JSON

```
{
  "book": {
        "type": "textbook",
        "pages": "256",
        "title": "Programming Pearls 2nd Edition",
        "description": "The first edition of Programming Pearls was
one of the most influential books I read early in my career...",
        "rating": "4.5",
        "coverType": "paperback",
        "genre": "Computer Science",
        "author": "Jon Bentley",
        "publisher": "Addison-Wesley Professional",
        "copyright": "1999"
        }
  }
```

JSON的背景

使用上面的 ~~XML~~ 和 JSON 文件分别运行解析测试10,000,000次。结果并不令人惊讶，解析和转换 JSON 成一个Java对象的速度比 ~~XML~~ 解析速度提高了30%,占用空间少30%。这些结果似乎和多数开发社区对两种格式的看法一样。

所以，换用 JSON 处理数据在性能上可以有不小的提升，而且还会减少空间的占用。

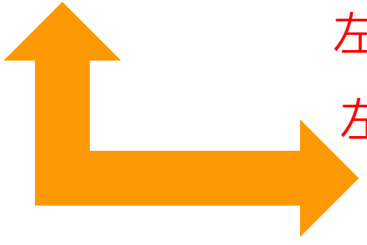
要NoSQL与要ACID

传统用户数据场景

- 强一致性
- 分段进行
- 规范统一

移动互联网数据场景

- 灵活构建
- 整体存储
- 分段展现



左边Insert，右边save

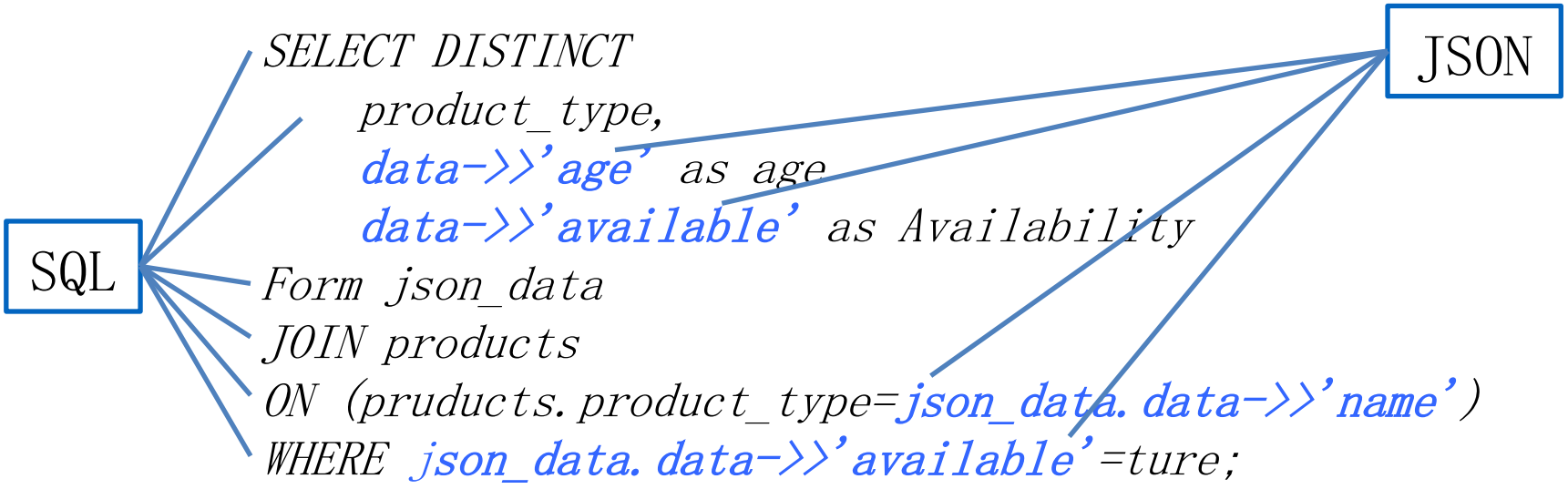
左边Select，右边find

码农就是996

终生无缘007

SQL + NoSQL(JSON) = NewSQL

```
CREATE TABLE json_data (data jsonb);
INSERT INTO json_data VALUES ('{"name":"xxx","age":"28", "available":true}');
```

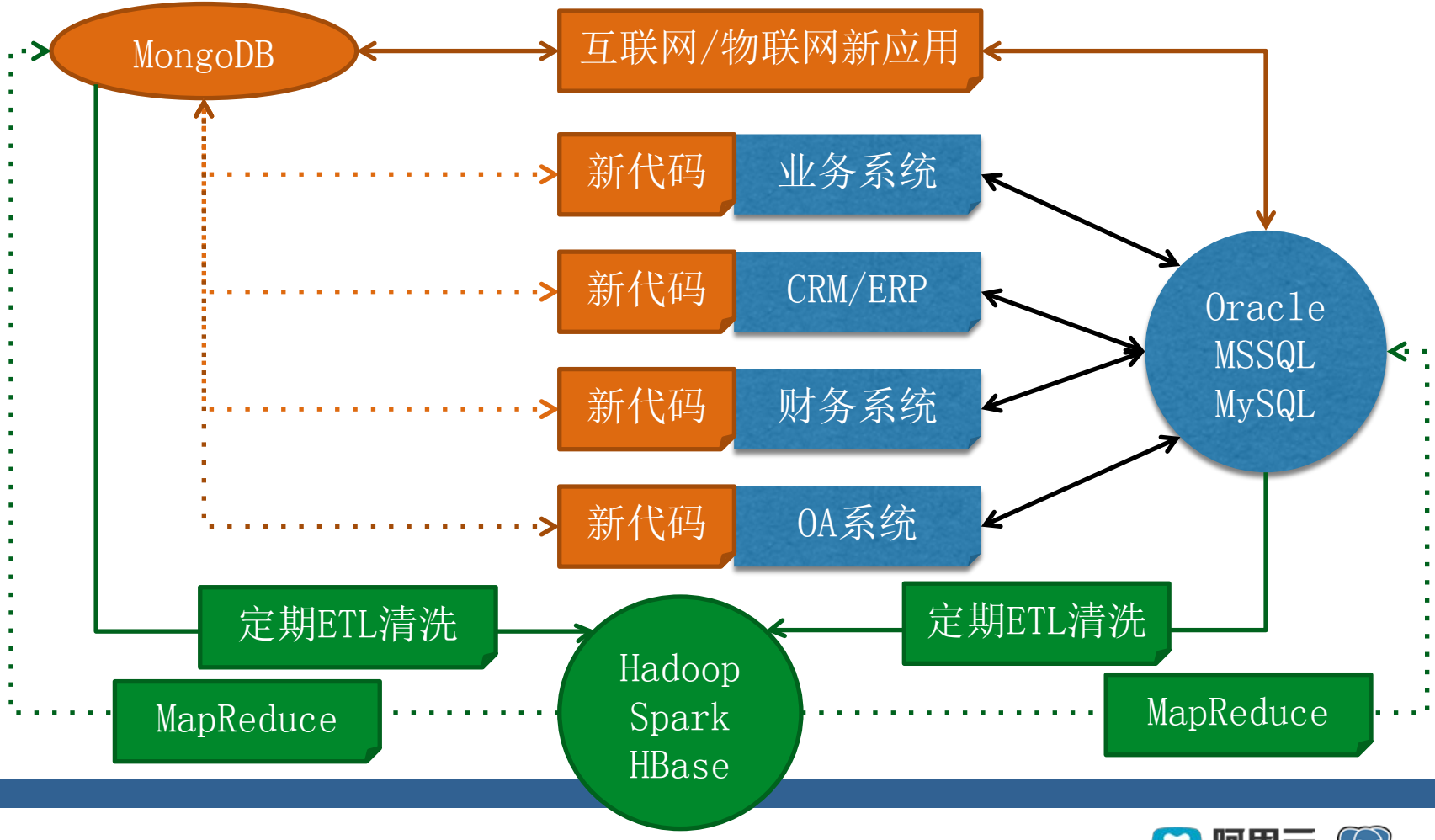


Postgres: JSON in SQL

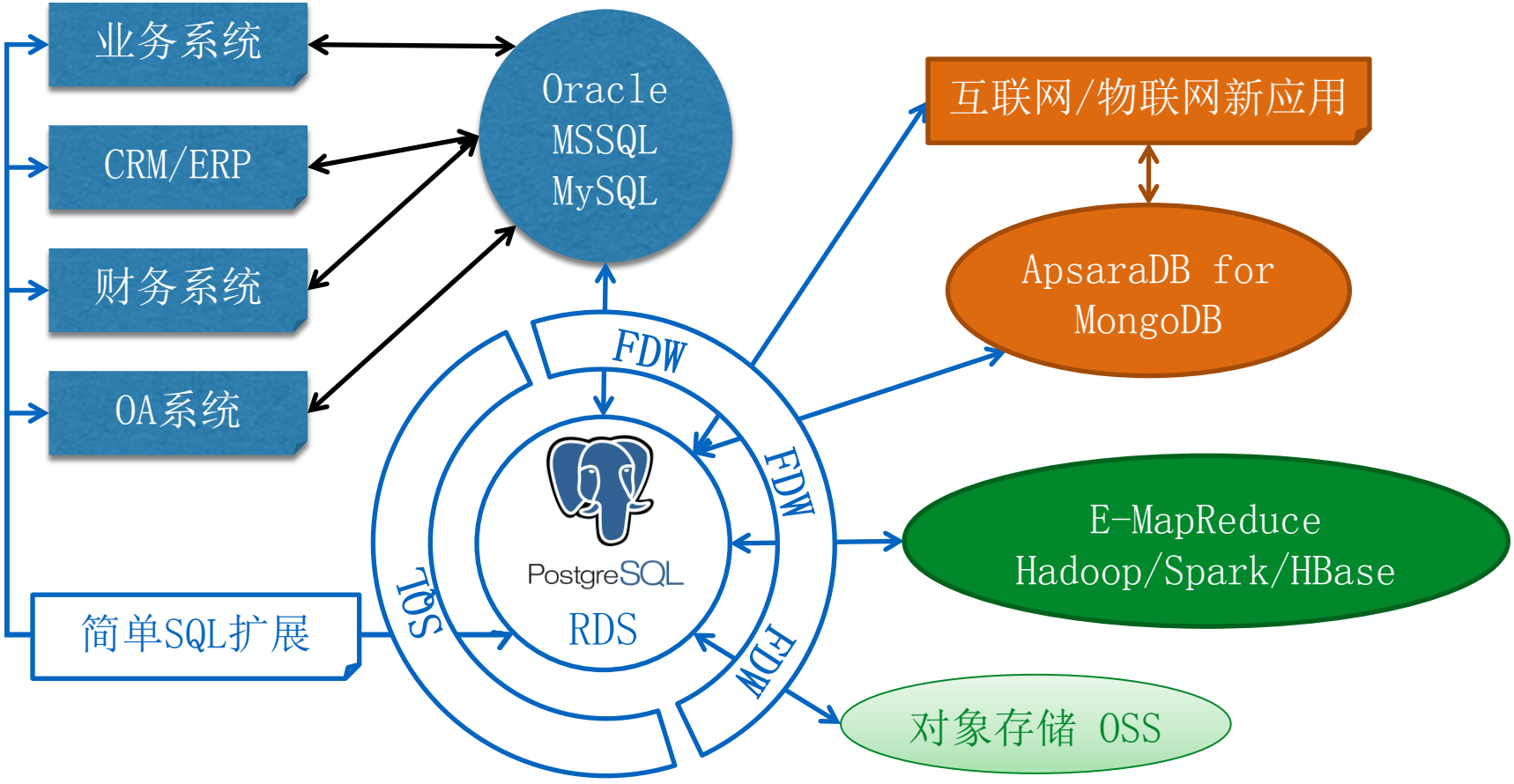
```
// require the Postgres connector
var pg = require("pg");

// connection to local database
var conString = "pg://postgres:password@localhost:5432/nodetraining";

var client = new pg.Client(conString);
client.connect();

// initiate the sample database
 client.query("CREATE TABLE IF NOT EXISTS emps(data jsonb)");
 client.query("TRUNCATE TABLE emps;");
 client.query('INSERT INTO emps VALUES($JSON$ {"firstname": "Ronald" , "lastname":"McDonald" }$JSON$)')
 client.query('INSERT  INTO emps values($JSON$ {"firstname": "Mayor", "lastname": "McCheese"}$JSON$)')

// run SELECT query
 client.query("SELECT * FROM emps",function(err,result){
     console.log("Test Output of JSON Result Object");
     console.log(result);
     console.log("Parsed rows");

// parse the result set
    for (var i = 0; i< result.rows.length ; i++ ){
        var data = JSON.parse(result.rows[i].data);
        console.log("First Name => "+ data.firstname + "\t| Last Name => " + data.lastname);
    }
 client.end();
})
```

PostgreSQL

借助FDW协助企业实现“敏捷项目”转型



借助FDW协助企业实现“敏捷项目”转型



Play with MongoDB

```
CREATE EXTENSION mongo_fw;

CREATE SERVER mongo_server
    FOREIGN DATA WRAPPER mongo_fdw
    OPTIONS (address 'dds-bp15bd497517f8a41.mongodb.rds.aliyuncs.com', port '3717');

CREATE USER MAPPING FOR postgresql
    SERVER mongo_server
    OPTIONS (username 'mongo', password 'mongo');

CREATE FOREIGN TABLE mongo_data(
    name text, type text, status text)
    SERVER mongo_server
     OPTIONS ( database 'apsaradb',  collection 'json_tables');
```

Play with MongoDB

```
SELECT * FROM mongo_data WHERE status='商业化' or status='邀测中' limit 10;

name | type | status
 ------------+-------+-------
ApsaraDB for RDS(MySQL) | RDS | 商业化
ApsaraDB for RDS(SQL Server) | RDS | 商业化
ApsaraDB for RDS(PostgreSQL) | RDS | 商业化
ApsaraDB for RDS(PPAS) | RDS | 商业化
ApsaraDB for MongoDB | NoSQL | 商业化
ApsaraDB for Redis | NoSQL | 商业化
ApsaraDB for Memcache | NoSQL | 商业化
ApsaraDB for PetaData | Distributed | 邀测中
ApsaraDB for OcenBase | Distributed | 邀测中
(9 rows)
```

Play with MongoDB

```
INSERT INTO mongo_data(name, type, status)  VALUES('ApsaraDB for Greenplum',
'Distributed', '邀测中');

SELECT* FROM mongo_data WHERE name='ApsaraDB for Greenplum';
 name | type | status
 ----------------+-------+-------
 ApsaraDB for Greenplum | Distributed | 邀测中

UPDATE mongo_data SET status='即将公测'  WHERE name='ApsaraDB for Greenplum';
SELECT * FROM mongo_data WHERE status='即将公测';
 name | type | status
 ----------------+-------+-------
 ApsaraDB for Greenplum | Distributed | 即将公测
```

FDW 还能支持什么数据源

Specific SQL Database Wrappers

Data Source	Type	Licence	Code	Install	Doc
PostgreSQL	Native	PostgreSQL	git.postgresql.org		documentation
Oracle	Native	PostgreSQL	github	PGXN	website
MySQL	Native		github	PGXN	example
MySQL	Native		github	PGXN	example
Informix	Native	PostgreSQL	github		
Firebird	Native		github	PGXN	
SQLite	Native		github		
Sybase / MS SQL Server	Native		github	PGXN	
MonetDB	Native		github		

FDW 还能支持什么数据源

NoSQL Database Wrappers

Data Source	Type	Licence	Code	Install	Doc	Notes
Cassandra2	Native		Github			
Cassandra	Multicorn	PostgreSQL	Github			
CouchDB	Native	PostgreSQL	Github	PGXN		Original version
CouchDB	Native	PostgreSQL	Github			golgauth version (9.1 - 9.2+ compatible)
Kyoto Tycoon	Native	MIT	Github			
MongoDB	Native	GPL3+	Github	PGXN	README	EDB version
MongoDB	Multicorn		Github			
MongoDB	Multicorn		Github			Yet Another Postgres FDW for MongoDB
Neo4j	Native	?	Github			
Quasar	Native	Apache	Github			
Redis	Native	PostgreSQL	Github			
Redis	Native	BSD	Github			
RethinkDB	Multicorn	MIT	Github		blog	
Riak	Multicorn		Github			
WhiteDB	Native	MIT	Github			

FDW 还能支持什么数据源

File Wrappers

Data Source	Type	Licence	Code	Install	Doc
CSV	Native	PostgreSQL			documentation
CSV	Multicorn	PostgreSQL	GitHub	PGXN	documentation
CSV / Text Array	Native		GitHub		How to
CSV / Fixed-length	Native		GitHub		
CSV / gzipped	Multicorn		GitHub		
Compressed File	Native		GitHub		
Document Collection	Native	PostgreSQL	GitHub		wiki
JSON	Native	GPL3	GitHub		Example
Multi-File	Multicorn	PostgreSQL	GitHub	PGXN	doc
Multi CDR	Native	PostgreSQL	GitHub	PGXN	
pg_dump	Native	New BSD	GitHub		
TAR Files	Native		GitHub		
XML	Multicorn	PostgreSQL	GitHub	PGXN	
ZIP Files	Native		GitHub		

FDW 还能支持什么数据源

Specific Web Wrappers

Data Source	Type	Licence
Database.com	Multicorn	BSD
Dun & Badstreet	Multicorn	
DynamoDB	Multicorn	GPL
Facebook	Multicorn	
Fixer.io	based on www_fdw	
Google	Multicorn	PostgreSQL
Heroku dataclips	Native	
Mailchimp	Multicorn	PostgreSQL
Parse	Multicorn	MIT
S3	Native	
S3CSV	Multicorn	GPL 3
Twitter	Native	
Treasure Data	Multicorn	Apache
Google Spreadsheets	Multicorn	MIT

Big Data Wrappers

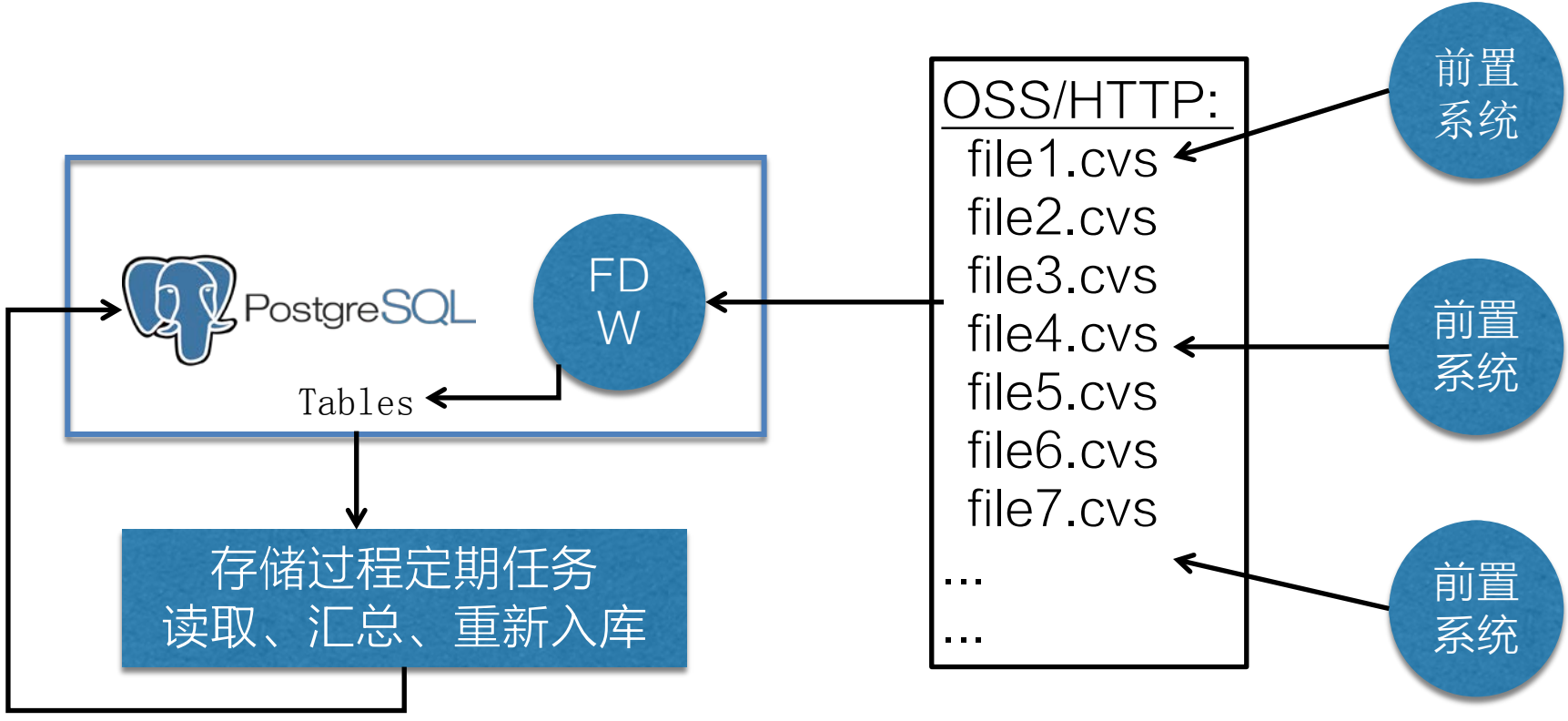
Data Source	Type	Licence
Elastic Search	Multicorn	
file_fdw-gds (Hadoop)	Native	
Hadoop	Native	
HDFS	Native	
Hive	Multicorn	
Hive / ORC File	Native	
Impala	Native	BSD

https://wiki.postgresql.org/wiki/Foreign_data_wrappers

提　纲

- Why Postgres
- Why FDW
- FDW: SQL Everything加速传统行业转型
- FDW: 金融报文处理
- FDW: 物联网数据整合
- FDW: 解决企业并购重组问题

FDW：金融报文处理


PostgreSQL
FD
W
Tables
存储过程定期任务
读取、汇总、重新入库
OSS/HTTP:
file1.cvs
file2.cvs
file3.cvs
file4.cvs
file5.cvs
file6.cvs
file7.cvs
...
...
前置
系统
前置
系统
前置
系统

FDW实现OSS操作

```
# 创建插件
create extension oss_fdw;

# 创建server，用于定位到OSS中的数据位置
CREATE SERVER ossserver FOREIGN DATA WRAPPER oss_fdw OPTIONS
        (host 'oss-cn-hangzhou-zmf.aliyuncs.com' , id 'xxx', key 'xxx',bucket 'mybucket');

# 创建基于OSS文件目录的名部表，将目录下所有文件的数据以表的形式展现
CREATE FOREIGN TABLE ossexample
       (date text, time text, open float, high float, low float, volume int)
        SERVER ossserver OPTIONS
       (dir 'osstest/', delimiter ',' , format 'csv', encoding 'utf8', PARSE_ERRORS '100');

# 通过表的方式对OSS中的数据进行访问
SELECT * FROM ossexample;
```

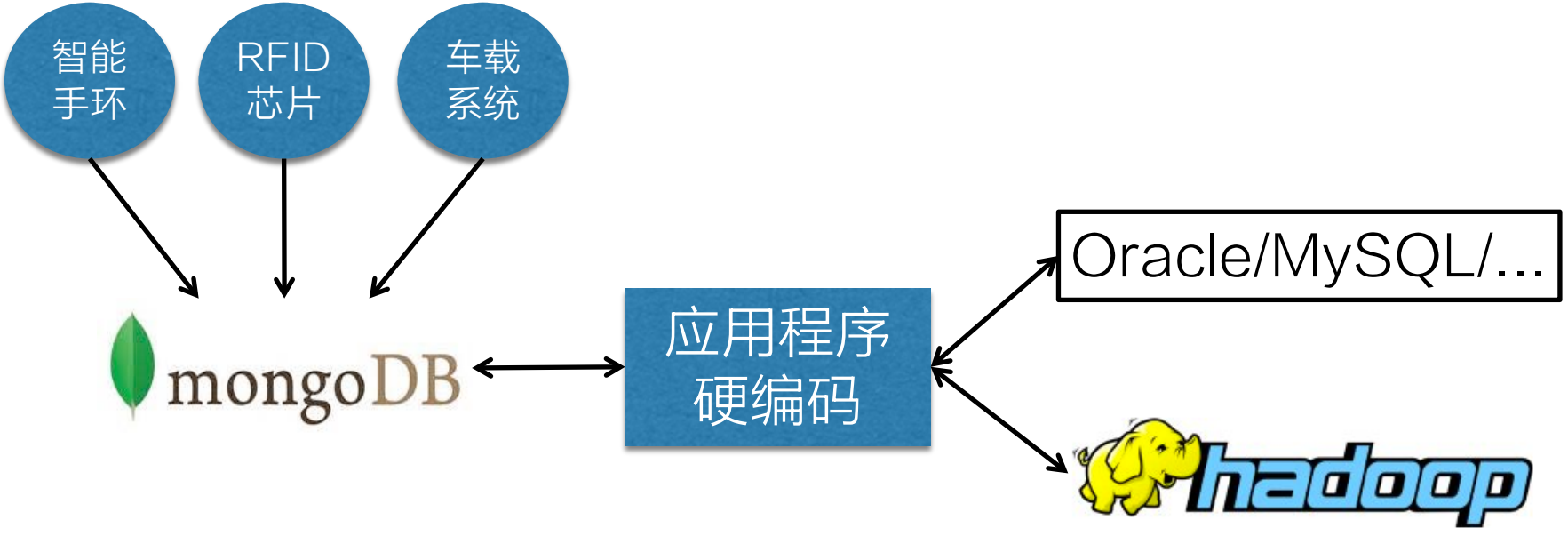
FDW实现OSS操作

```
# 创建表，将OSS中的数据写入到PostgreSQL数据库中
create table example (date text, time text, open float, high float, low float, volume int);

# 数据并行的从 ossexample 装载到 example 中
insert into example select * from ossexample;

# 可以看到 oss_fdw 能够正确估计 oss 上的文件大小，正确的规划查询计划。
explain insert into example select * from ossexample;
                          QUERY PLAN
--------------------------------------------------------------------------
 Insert on example  (cost=0.00..1.60 rows=6 width=92)
   ->  Foreign Scan on ossexample  (cost=0.00..1.60 rows=6 width=92)
         Foreign OssFile: osstest/example.csv.0
         Foreign OssFile Size: 728
(4 rows)
```

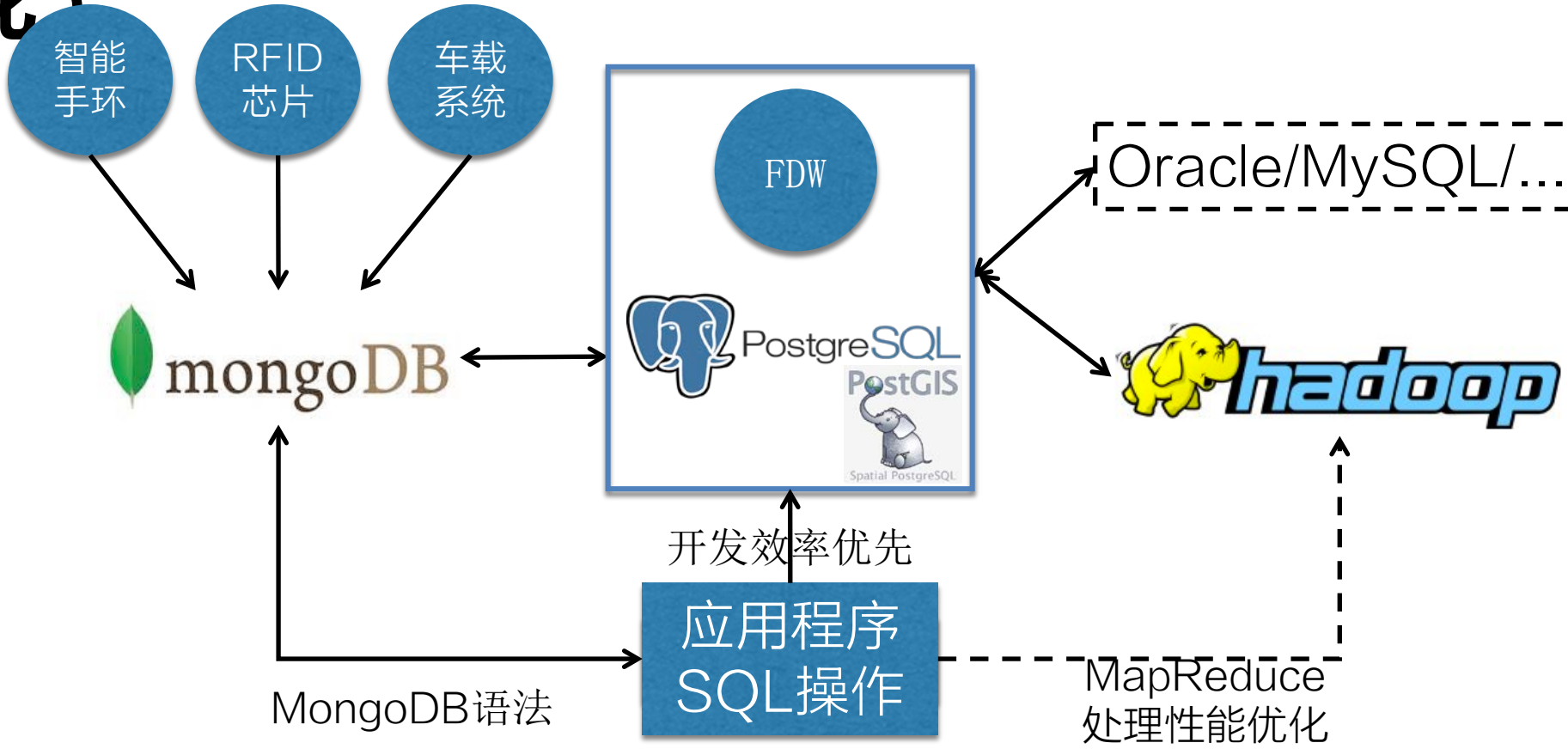
提　纲

- Why Postgres
- Why FDW
- FDW: SQL Everything加速传统行业转型
- FDW: 金融报文处理
- FDW: 物联网数据整合
- FDW: 解决企业并购重组问题

FDW：物联网数据整合（现有环境）



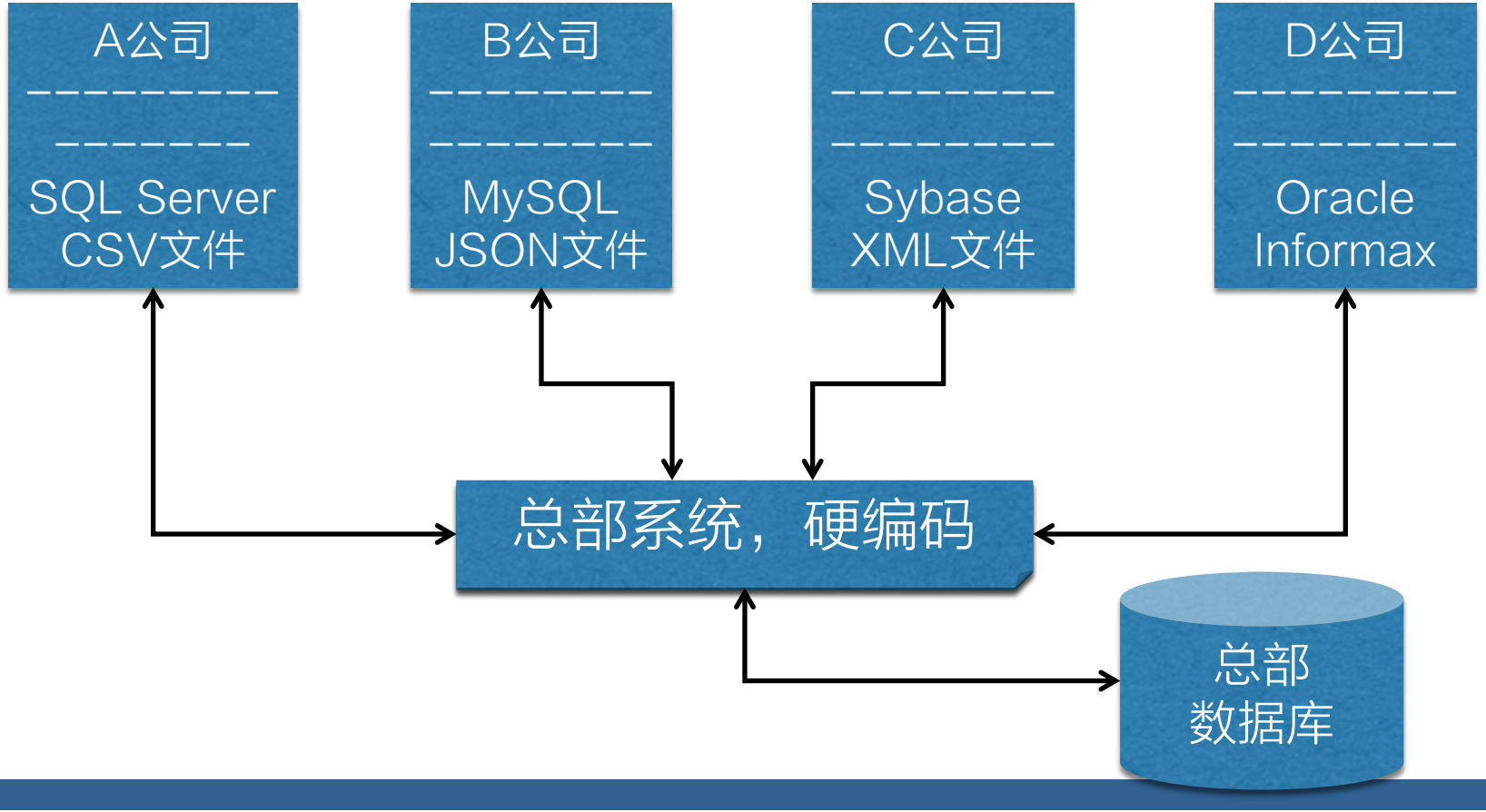
FDW：物联网数据整合（开发效率&性能优化）



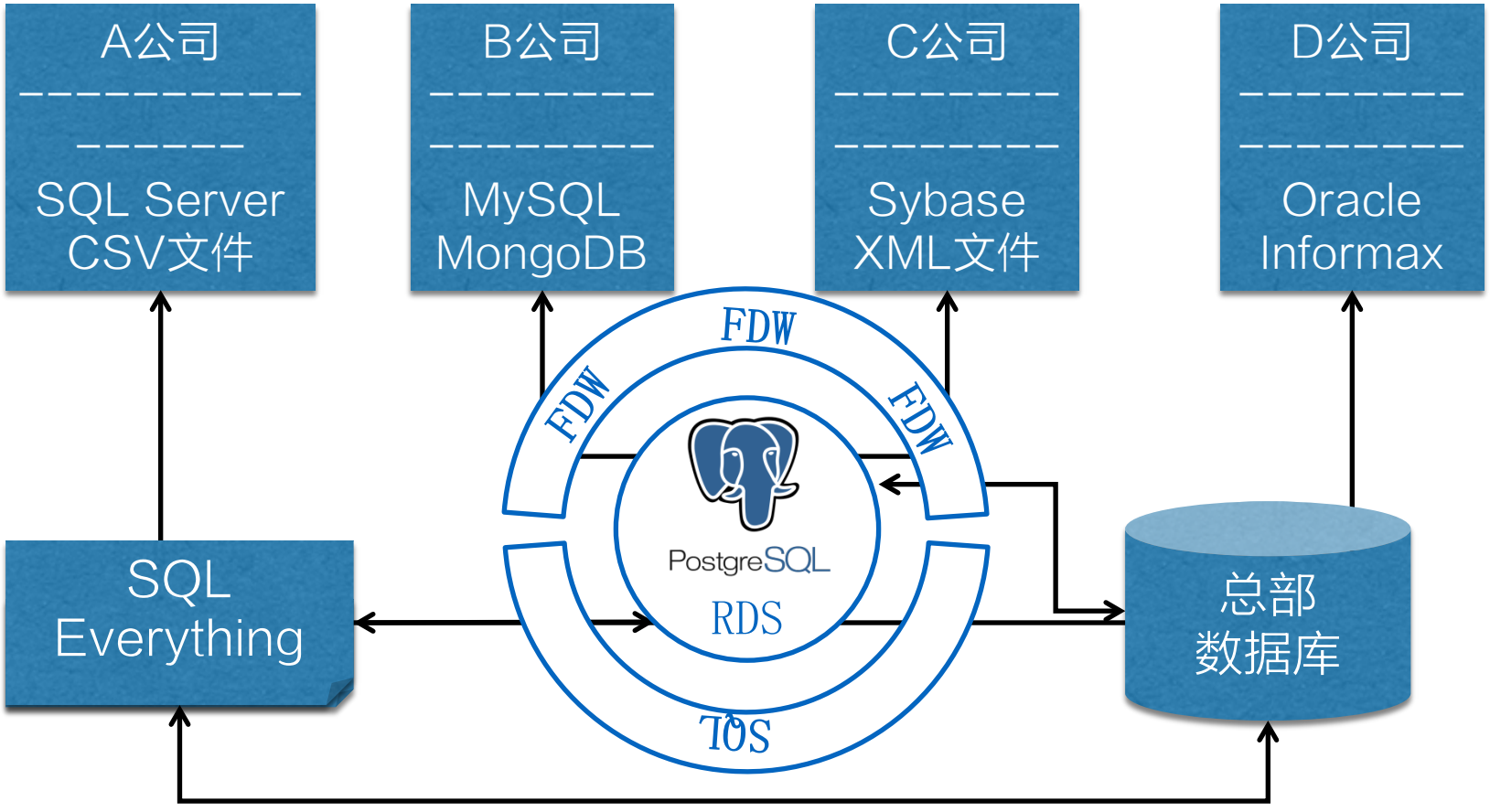
提　纲

- Why Postgres
- Why FDW
- FDW: SQL Everything加速传统行业转型
- FDW: 金融报文处理
- FDW: 物联网数据整合
- FDW: 解决企业并购重组问题

FDW：解决企业并购重组问题



FDW：解决企业并购重组问题



阿里云 aliyun.com

关注微博
(@萧少聪Postgres)

关注云栖社区
微信公众号