xHarbour
eXtended xBase Compiler

Implementing SQLRDD

Why should you migrate to SQL ?

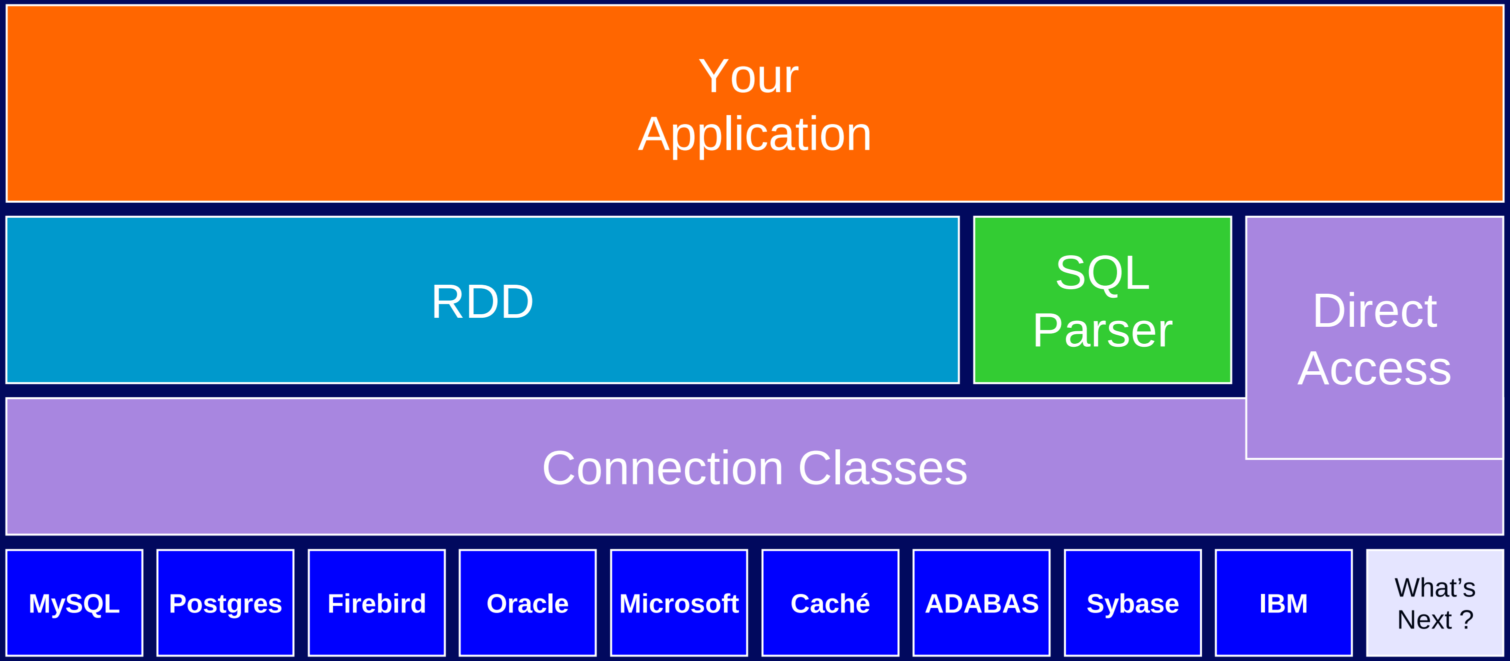
- Security
- Physical integrity
- Logical application integrity (transactional control)
- Speed should not be the main reason!

First, let's say it correctly...

▪ file, database	**TABLE**
▪ Field	**COLUMN**
▪ Record	**LINE**
▪ Directory	**DATABASE**

- **DML** – Data manipulation language (seek, replace, skip, etc..)
- **DDL** – Data definition language (dbCreate(), INDEX, etc.)

SQLRDD layers



xHarbour eXtended xBase Compiler

RDD (Replaceable Database Driver)

- DBFCDX compatible
- DDL and DML support
- Translate xBase (ISAM) to SQL
  - Cache Workareas
  - Paging Workareas
- Connect to databases through “Connection Classes”

XHarbour
eXtended xBase Compiler

Connection Classes

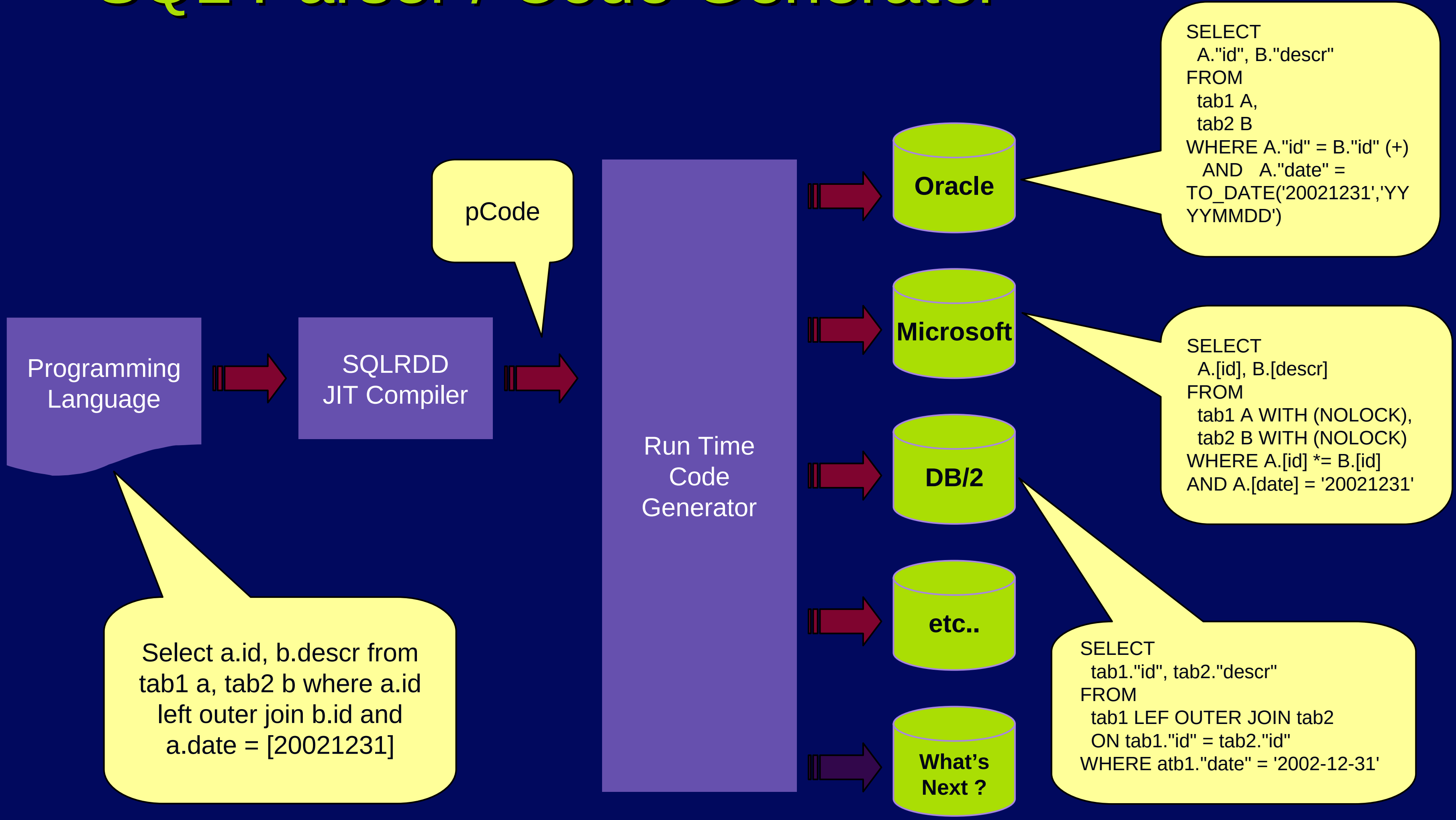
- Ensemble of classes that provides database access
- Direct record set manipulation
- Provides direct database access to applications, but with database’s suitable SQL dialect (not portable)

SQL Parser

- Provides a “natural”, database independent SQL language
- Compiles the natural SQL and generates an SQL pCode
- Generates database specific SQL code based in SQL pCode
- Processes only DML at this time (SELECT, INSERT, UPDATE, DELETE)

xHarbour
eXtended xBase Compiler

SQL Parser / Code Generator



```
Select a.id, b.descr from
tab1 a, tab2 b where a.id
left outer join b.id and
a.date = [20021231]
```

```
SELECT
  A."id", B."descr"
FROM
  tab1 A,
  tab2 B
WHERE A."id" = B."id" (+)
   AND   A."date" =
TO_DATE('20021231','YY
YYMMDD')
```

```
SELECT
  A.[id], B.[descr]
FROM
  tab1 A WITH (NOLOCK),
  tab2 B WITH (NOLOCK)
WHERE A.[id] *= B.[id]
AND A.[date] = '20021231'
```

```
SELECT
  tab1."id", tab2."descr"
FROM
  tab1 LEF OUTER JOIN tab2
  ON tab1."id" = tab2."id"
WHERE atb1."date" = '2002-12-31'
```

XHarbour
eXtended xBase Compiler

Differentials

- Only tool in the market that allows real portability to many different databases (it has no similar in any other language)
- Does not need any middleware or server side software
- Creates royalty free applications
- Very few changes in source code
- The wider range of supported databases
- Uses database native data types and indexes
- Can share database with other languages and applications

Migration methodology

1. Instant migration
2. Gaining Performance
3. Fine tuning (optional)

xHarbour eXtended xBase Compiler

Step 01: Instant migration

- Migrate from Clipper to xHarbour, but still using DBF
- Run dbf2sql to migrate DBF structure and data to target database
- Add database connection to your main procedure
- Change table open to “VIA SQLRDD” or change default RDD - RDDSetDefault(“SQLRDD”)
- Replace file() with sr_file() where needed
- Add transactional control in strategic application points
- Basic database server setup

Results from step 01

- Screens and browses have a good performance
- Programs also have a good performance, with some localized slow points
- Usually, reports become slow
- Great application security and integrity improvement
- Applications are ready to be delivered to clients with “urgent integrity problems”
- Estimate time is 1 to 5 work days

Step 02: Gaining performance

- Change main reports to use SQL queries (you may use SQL Parser/Code Generator to have it portable)
- Change table open to SET AUTOPEN ON
- Adjust processing code where you find:
  - Seek/DoWhile< condition>/skip/EndDo, replace with UPDATE ... SET .. = .. WHERE <condition>
  - Summarizing loops, replace with SELECT ... WHERE <condition>

Results from step 02

- All application with good performance, and in some points, faster than DBF
- Application is ready to be delivered to clients in general
- Estimate time is 1 week to 3 months

Step 03: Fine tuning (optional)

- Rework old bad code
- Adopt server side filters
- Use exclusive SQLRDD techniques (not supported by other RDDs)
  - SR_SetlGoTopOnFirstInteract(.F.)
  - SR_SetGoTopOnScope(.F.)
  - Synthetic Indexes
- Add referential and relational integrity to database
- Tuning in database server made by a certified DBA (data base administrator)

Results from step 03

- Performance far better than DBF
- Application is ready to all terrain
- Incontestable application platform to most of the DBAs