Работа с реляционными БД в C++

Николай Гродзицкий

О чем пойдет речь?

О чем пойдет речь?

1. Родные клиенты для работы с БД

О чем пойдет речь?

1. Родные клиенты для работы с БД
2. “Сторонние” библиотеки для доступа к БД

О чем пойдет речь?

1. Родные клиенты для работы с БД
2. “Сторонние” библиотеки для доступа к БД
3. Что будет завтра?

Родные клиенты для работы с БД

ODBC

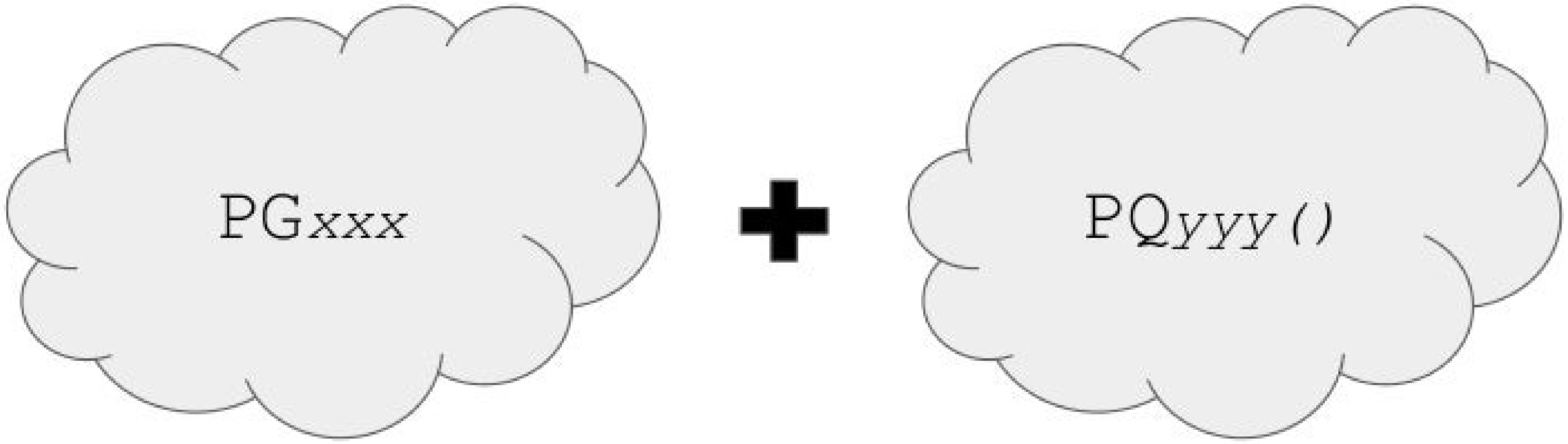
Родные клиенты для работы с БД

Родные клиенты для работы с БД

- http://postgresql.org/
- https://www.postgresql.org/docs/9.6/static/libpq.html
- https://www.postgresql.org/docs/9.6/static/ecpg.html
- https://github.com/jtv/libpqxx

Родные клиенты для работы с БД

libpq



Родные клиенты для работы с БД

libpqxx

- OOP
- ~~RAII~~
- exceptions
- type conversion

Родные клиенты для работы с БД

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int main() {
  EXEC SQL BEGIN DECLARE SECTION;
    char str[25];
    int i, count=1;
  EXEC SQL END DECLARE SECTION;

  EXEC SQL CONNECT TO some_db;

  EXEC SQL CREATE TABLE Foo(Item1int,Item2 text);

  EXEC SQL INSERT INTO My_Table VALUES(1,'txt1');
  EXEC SQL INSERT INTO My_Table VALUES(2,'txt1');

  EXEC SQL DECLARE CUR CURSOR FOR SELECT * FROM
My_Table;

   . . .
```

ECPG

```
#include <ecpglib.h>
#include <ecpgerrno.h>
#include <sqlca.h>

#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int main() {
/* exec sql begin declare section */
 char str [ 25 ] ;
 int i , count = 1 ;
/* exec sql end declare section */
  { ECPGconnect(__LINE__, 0, "some_db" , NULL, NULL , NULL, 0); }
  { ECPGdo(__LINE__, 0, 1, NULL, 0, ECPGst_normal, "create table
Foo ( Item1 int , Item2 text )", ECPGt_EOIT, ECPGt_EORT);}
  { ECPGdo(__LINE__, 0, 1, NULL, 0, ECPGst_normal, "insert into
Foo values ( 1 , 'txt1' )", ECPGt_EOIT, ECPGt_EORT);}
  { ECPGdo(__LINE__, 0, 1, NULL, 0, ECPGst_normal, "insert into
Foo values ( 2 , 'txt2' )", ECPGt_EOIT, ECPGt_EORT);}
  /* declare CUR cursor for select * from Foo */
  { ECPGdo(__LINE__, 0, 1, NULL, 0, ECPGst_normal, "declare CUR
cursor for select * from Foo", ECPGt_EOIT, ECPGt_EORT);}
```

Родные клиенты для работы с БД

- https://www.mysql.com/
- https://dev.mysql.com/doc/connectors/en/connector-c.html
- https://dev.mysql.com/doc/connectors/en/connector-cpp.html

stiffstream

Родные клиенты для работы с БД

MySQL Connector/C

`MYSQL_`*`xxx`* + `mysql_`*`yyy()`*

Родные клиенты для работы с БД

MySQL Connector/C++

- OOP
- ~~RAII~~
- exceptions
- type conversion

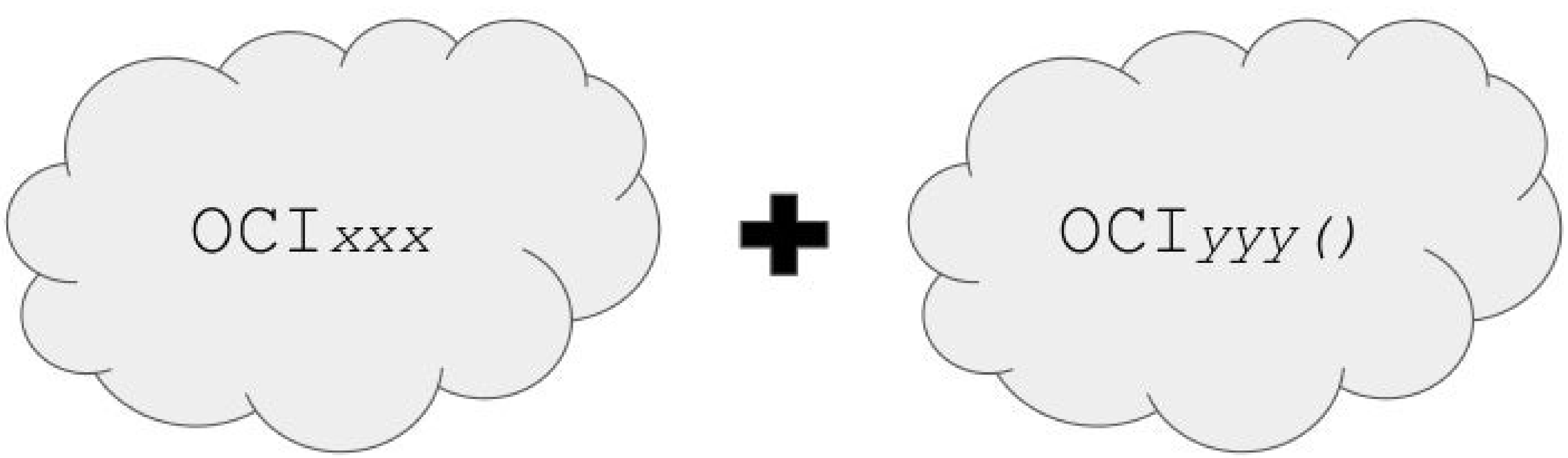
Родные клиенты для работы с БД

ORACLE

- https://www.oracle.com/database/index.html
- Instant-clients: http://www.oracle.com/technetwork/database/features/instant-client/index-097480.html
- OCI: http://docs.oracle.com/database/122/LNOCI/instant-client.htm#LNOCI-GUID-AAB0378F-2C7B-41EB-ACAC-18DD5D052B01
- OCCI: http://docs.oracle.com/database/122/LNCPP/installation-and-upgrading.htm#LNCPP002
- Pro*C: http://docs.oracle.com/database/122/LNPCC/introduction.htm#LNPCC3028

Родные клиенты для работы с БД

OCI



Родные клиенты для работы с БД

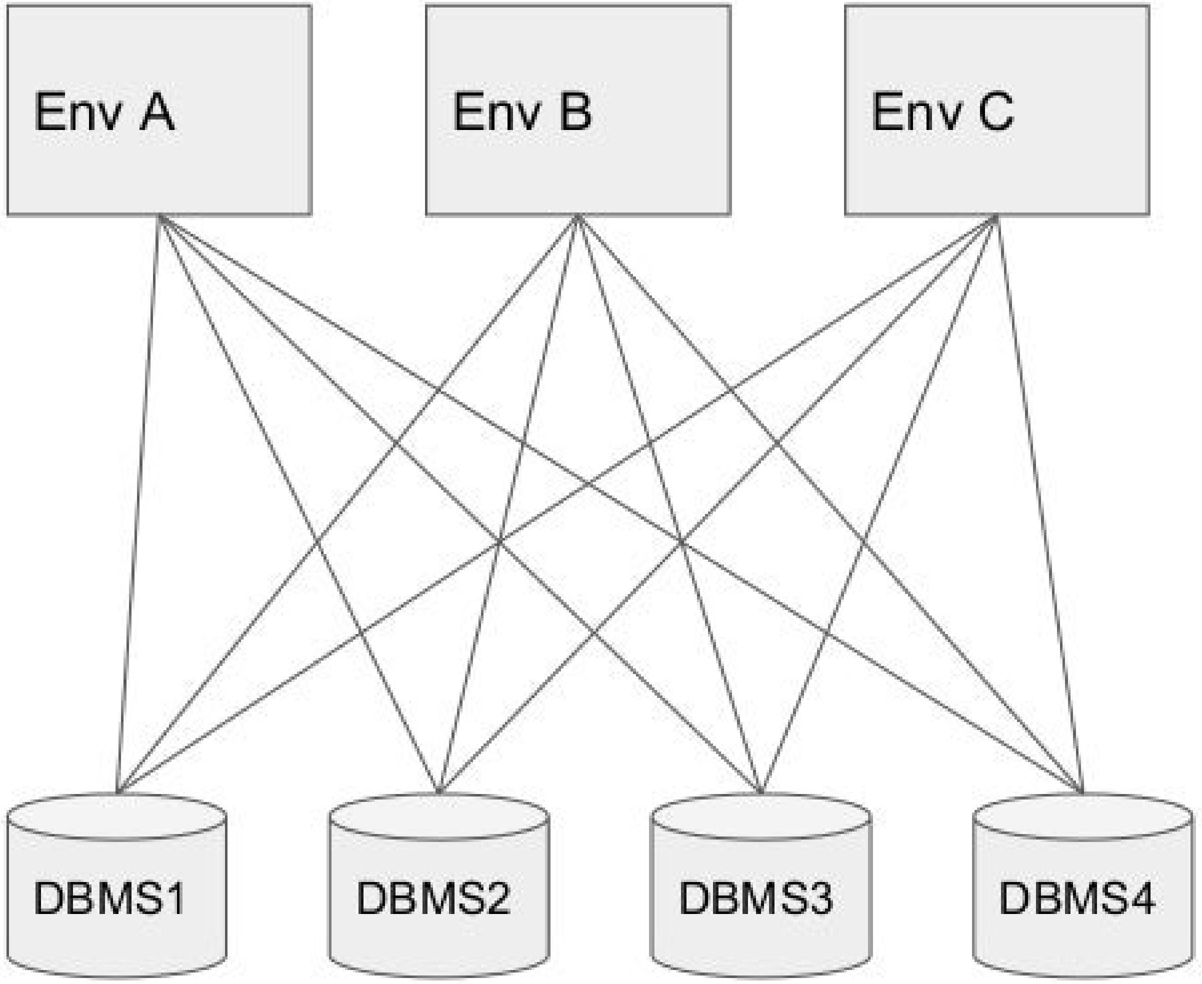
OCCI

- OOP
- ~~RAII~~
- exceptions

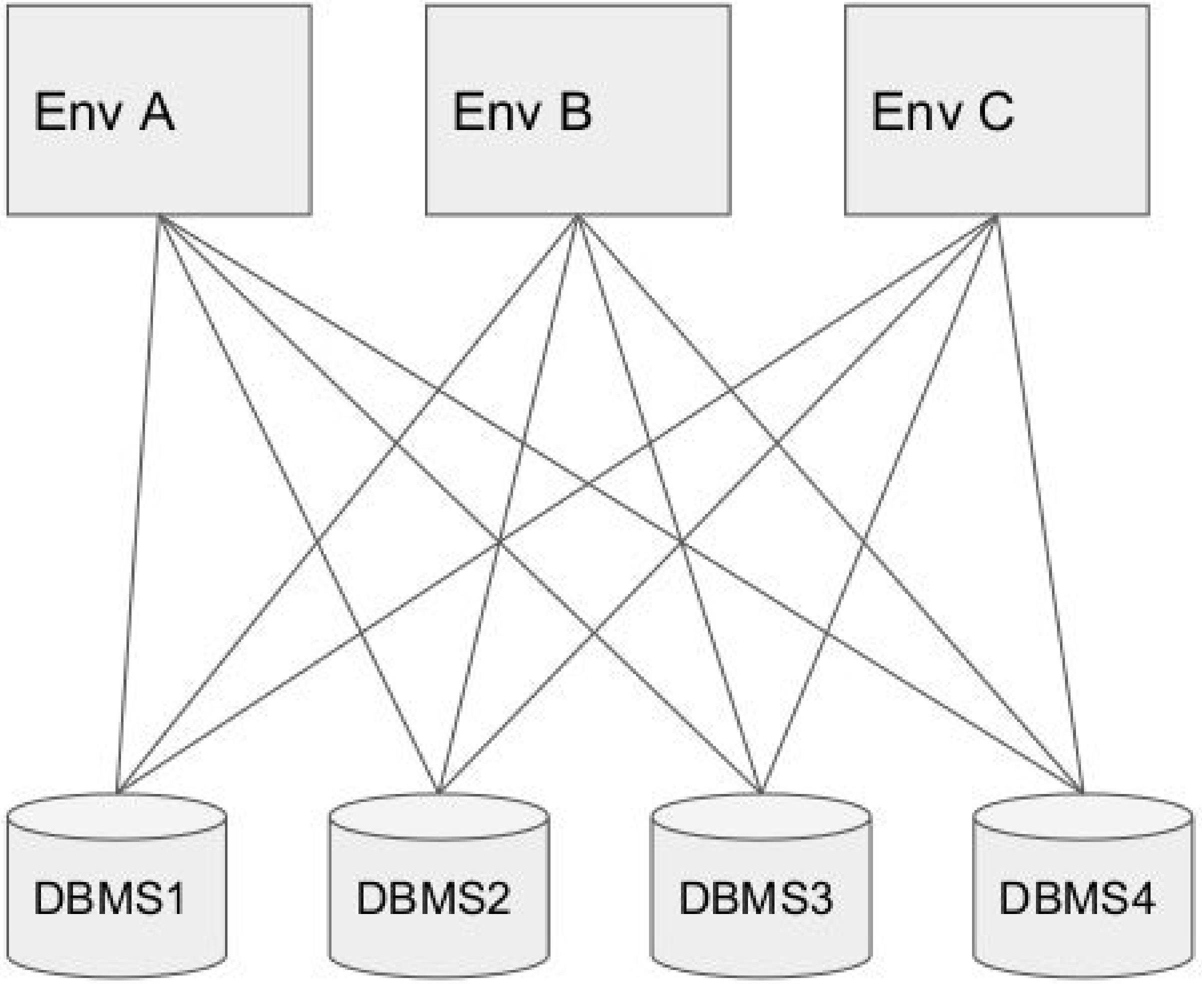
Родные клиенты для работы с БД

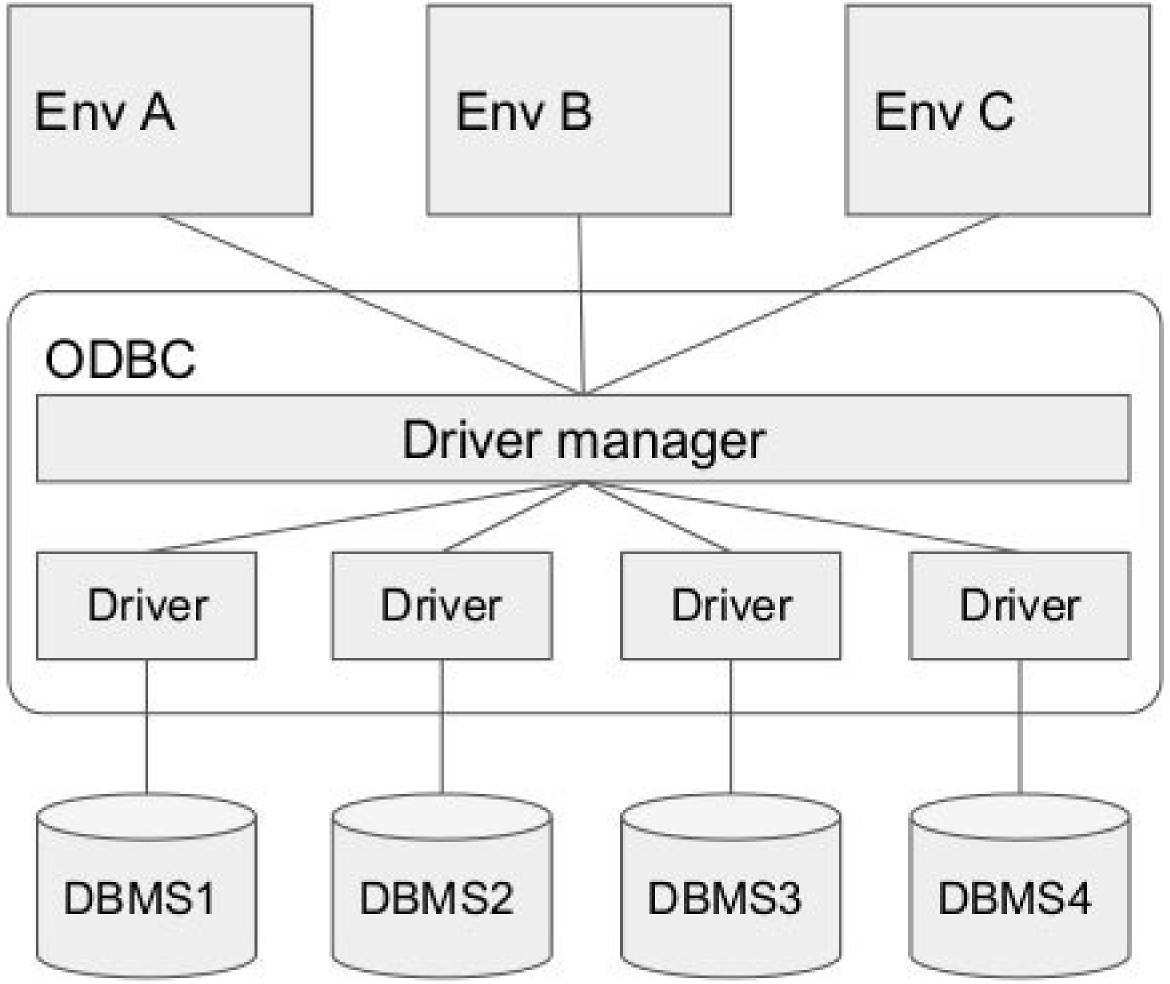
- http://www.microsoft.com/sqlserver/
- https://docs.microsoft.com/en-us/sql/relational-databases/native-client/sql-server-native-client-programming
- https://docs.microsoft.com/en-us/sql/odbc/reference/odbc-programmer-s-reference

ODBC

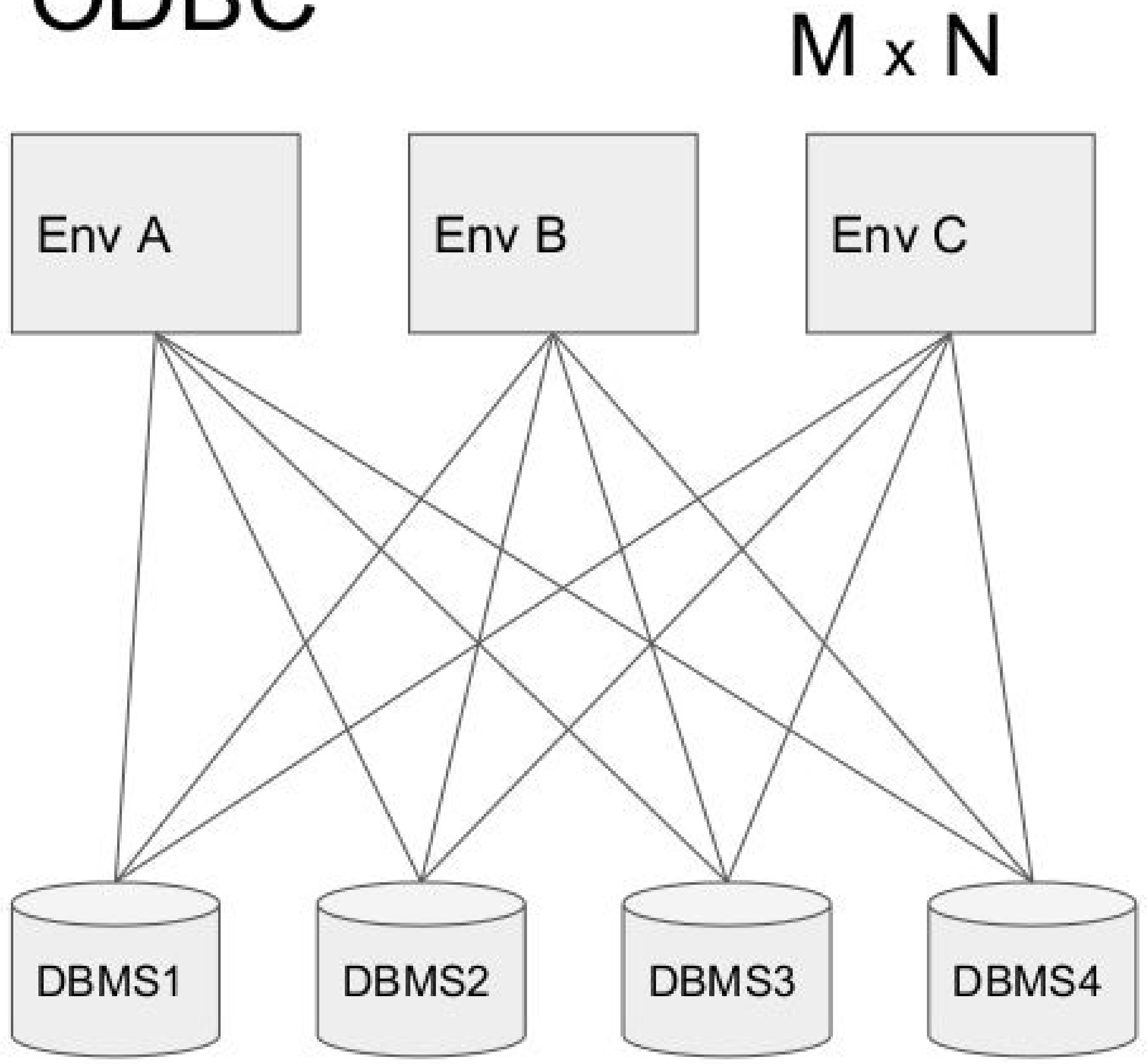


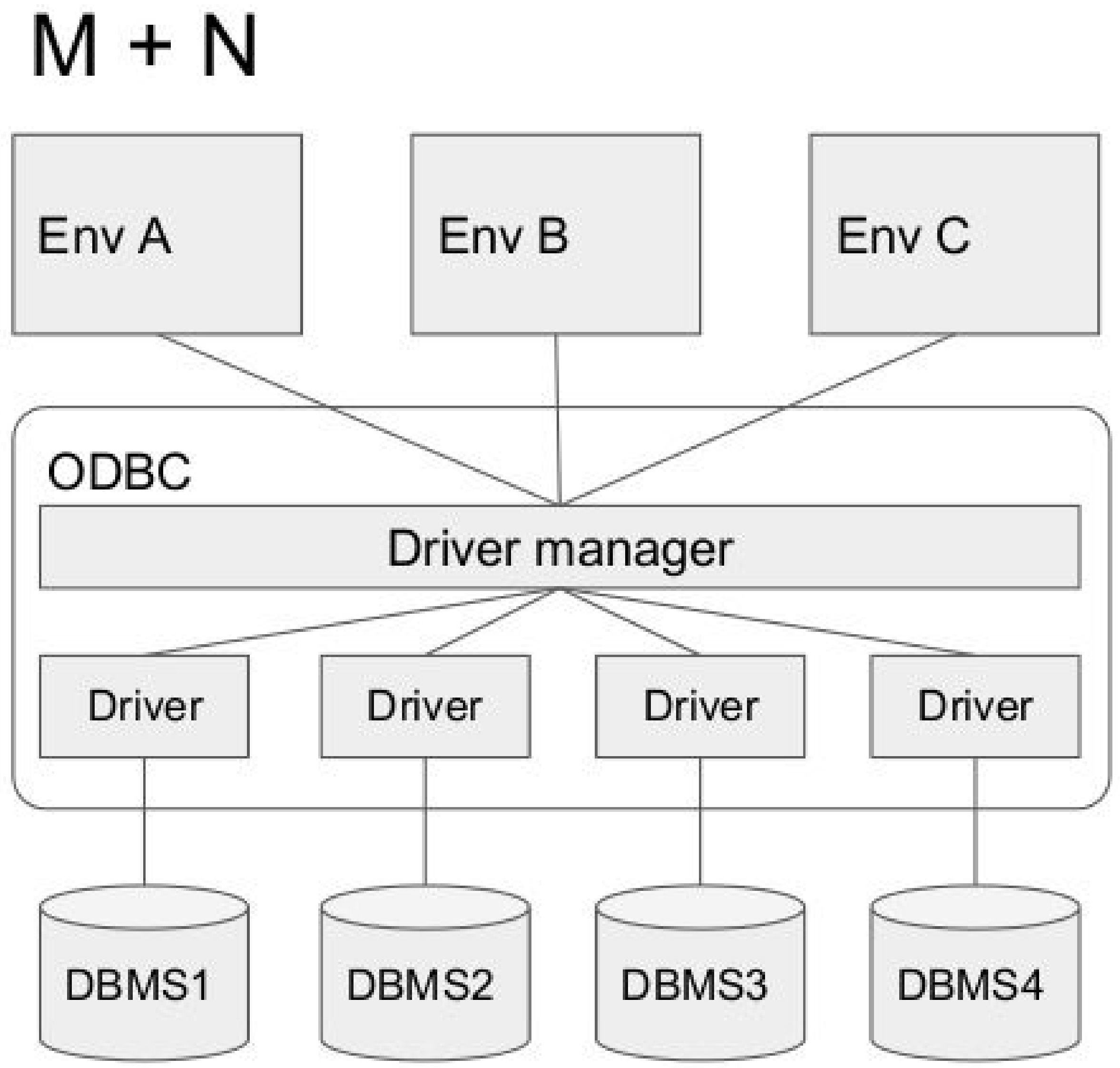
ODBC


Env A
Env B
Env C
DBMS1
DBMS2
DBMS3
DBMS4


Env A
Env B
Env C
ODBC
Driver manager
Driver
Driver
Driver
Driver
DBMS1
DBMS2
DBMS3
DBMS4

ODBC


M x N
Env A
Env B
Env C
DBMS1
DBMS2
DBMS3
DBMS4


M + N
Env A
Env B
Env C
ODBC
Driver manager
Driver
Driver
Driver
Driver
DBMS1
DBMS2
DBMS3
DBMS4

ODBC

API...

ODBC

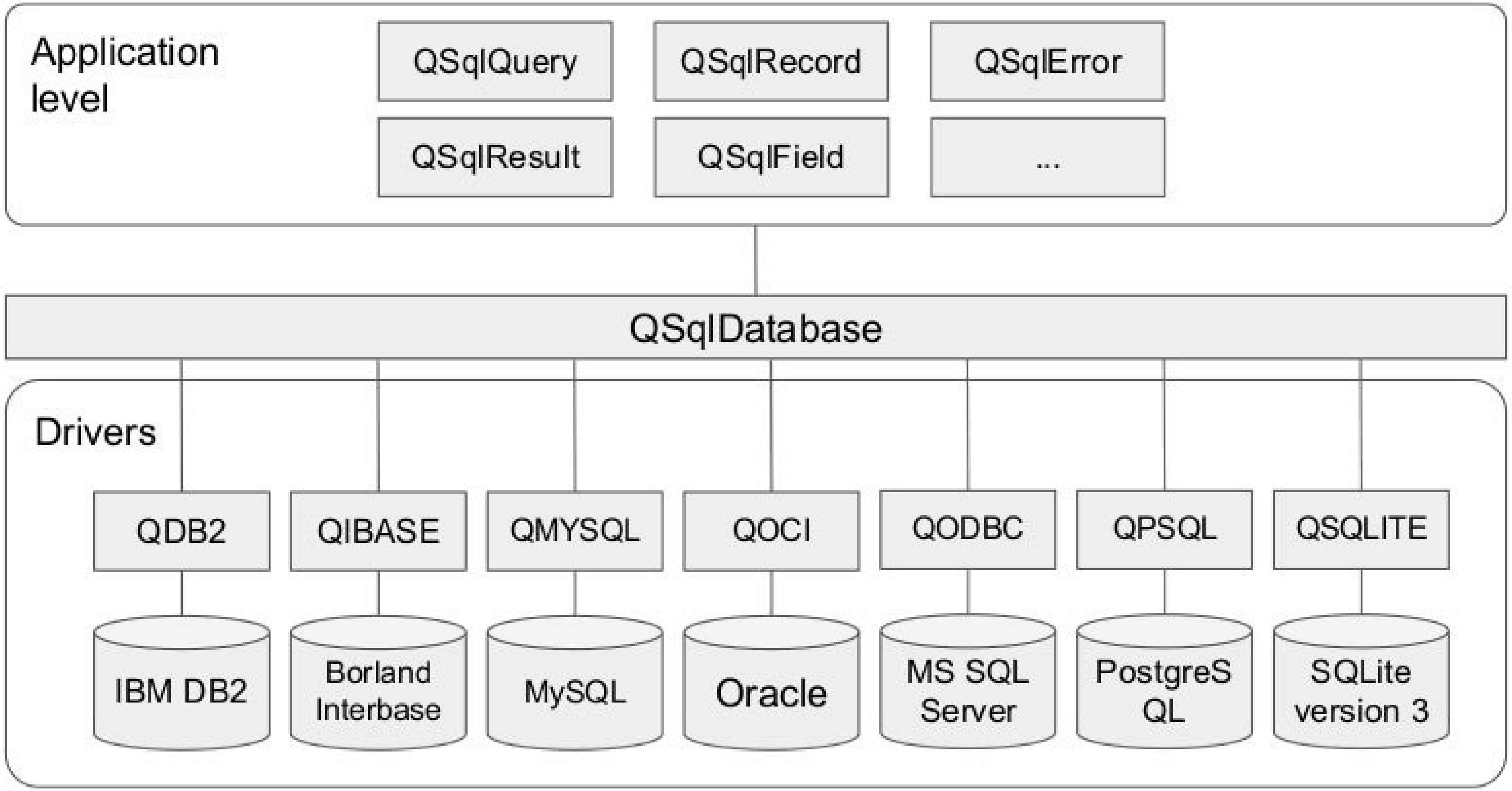
“Сторонние” библиотеки для доступа к БД

Что же есть готового?

Что же есть готового?

- В составе больших библиотек:
  - QtSQL http://doc.qt.io/qt-5/qtsql-index.html
  - Poco::Data https://pocoproject.org/docs/Poco.Data.html https://pocoproject.org/docs/00200-DataUserManual.html
- Обертки над родными клиентами и ODBC:
  - OTL http://otl.sourceforge.net/otl3_intro.htm
  - SOCI http://soci.sourceforge.net/index.html
  - Sqlpp11 https://github.com/rbock/sqlpp11
  - Sqlapi++ http://www.sqlapi.com/index.html (shareware)

QtSQL


Application level
QSqlQuery
QSqlRecord
QSqlError
QSqlResult
QSqlField
...
QSqlDatabase
Drivers
QDB2
QIBASE
QMYSQL
QOCI
QODBC
QPSQL
QSQLITE
IBM DB2
Borland Interbase
MySQL
Oracle
MS SQL Server
PostgreSQL
SQLite version 3

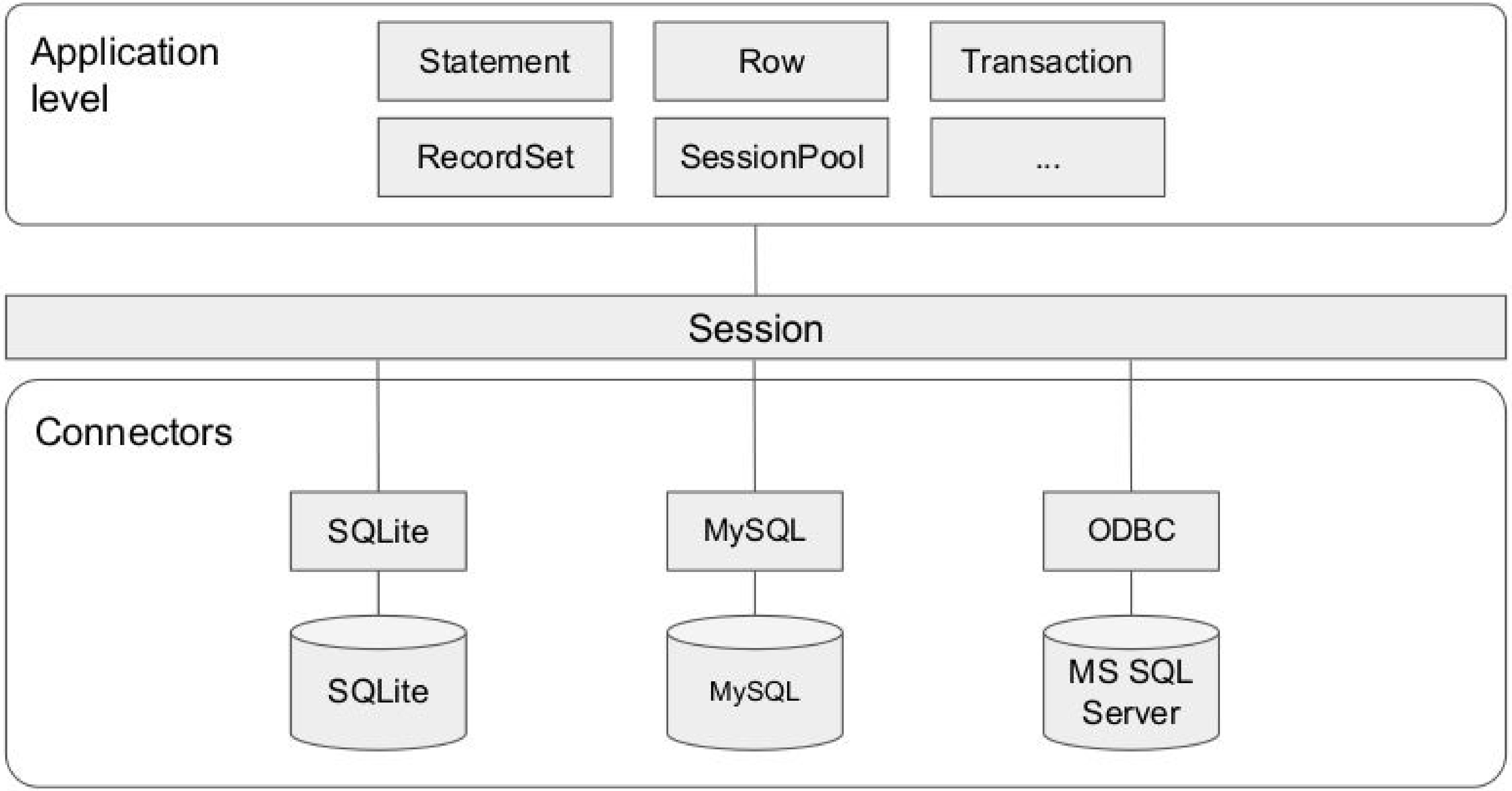
QtSQL

- Prepared statements;
- Transaction support;
- BLOB (не для всех СУБД);
- Binding: named, positional;
- Bulk IO.

QtSQL

```
QSqlDatabase db = QSqlDatabase::addDatabase("QMYSQL");
db.setHostName("xxx"); db.setDatabaseName("yyy");
assert( db.open("user", "pass") );

db.transaction();

QSqlQuery insert_query(db);
insert_query.prepare(
  "INSERT INTO project (id, name, employeeid) "
  "VALUES (201, 'Manhattan Project', ? )" );

QSqlQuery select_query("SELECT id FROM employee WHERE name = 'Albert'", db);

while( select_query.next() ){
  int employeeId = select_query.value(0).toInt();
  insert_query.addBindValue( employeeId );
  insert_query.exec();
}
db.commit();
```

Poco::Data


Application level
Statement
Row
Transaction
RecordSet
SessionPool
...
Session
Connectors
SQLite
MySQL
ODBC
SQLite
MySQL
MS SQL Server

Poco::Data

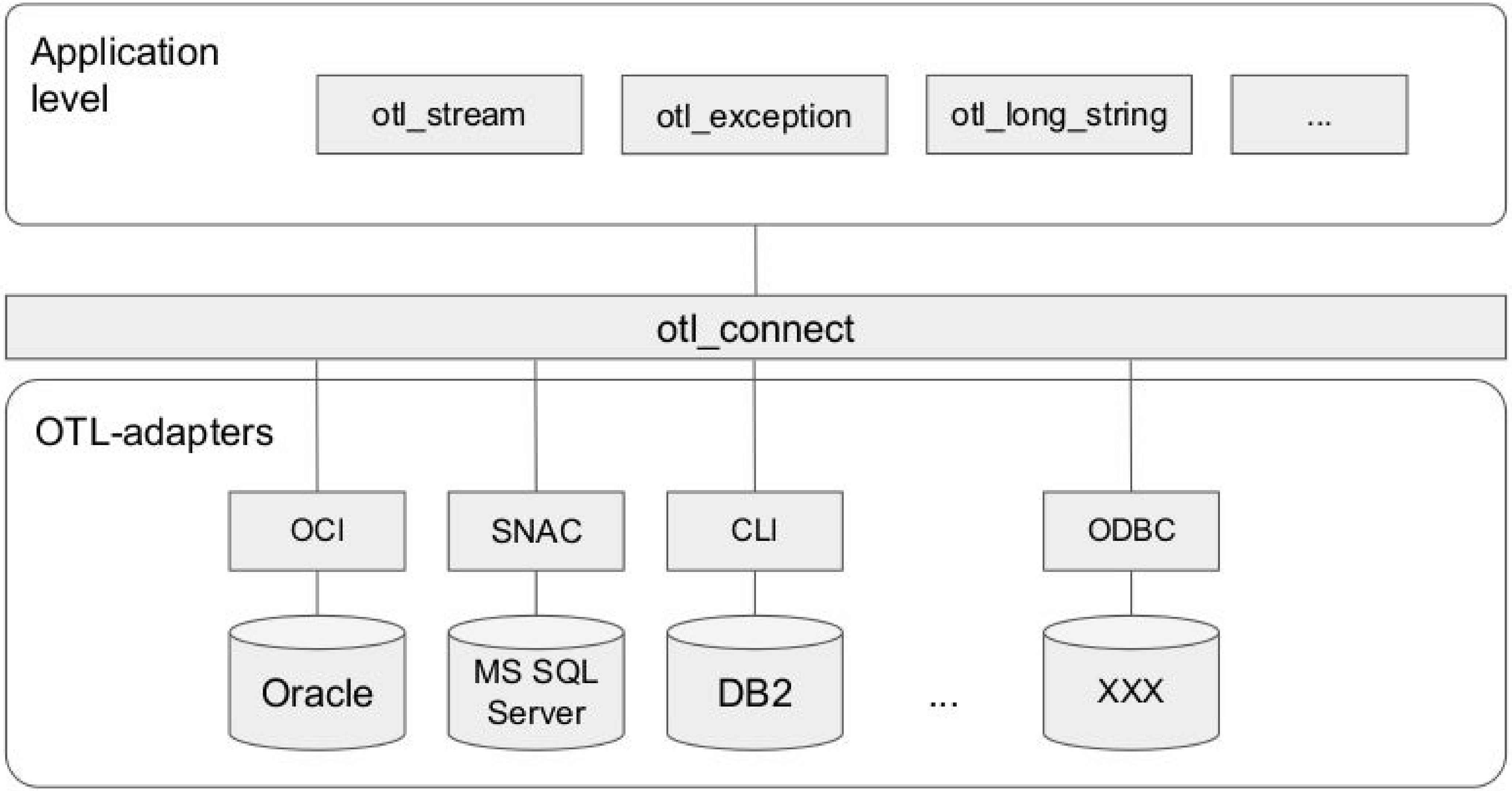
- Prepared statements;
- Transaction support;
- BLOB;
- Binding;
- Bulk IO;
- Complex Data Types.

stiffstream

Poco::Data

```
using namespace Poco::Data;
SQLite::Connector::registerConnector();
Session ses("SQLite", "sample.db");

Transaction trx( ses );

std::vector< int > employee_ids;
Statement select_query(session);
select_query << "SELECT id FROM employee WHERE name = 'Albert'" ,
                into(employee_ids, bulk(100) );

Statement insert_query( ses );
insert_query << "INSERT INTO project (id, name, employeeid) "
                "VALUES (201, 'Manhattan Project', ? )" ,
                use(employee_id, bulk(employee_id.size()));

trx.commit();
```

OTL

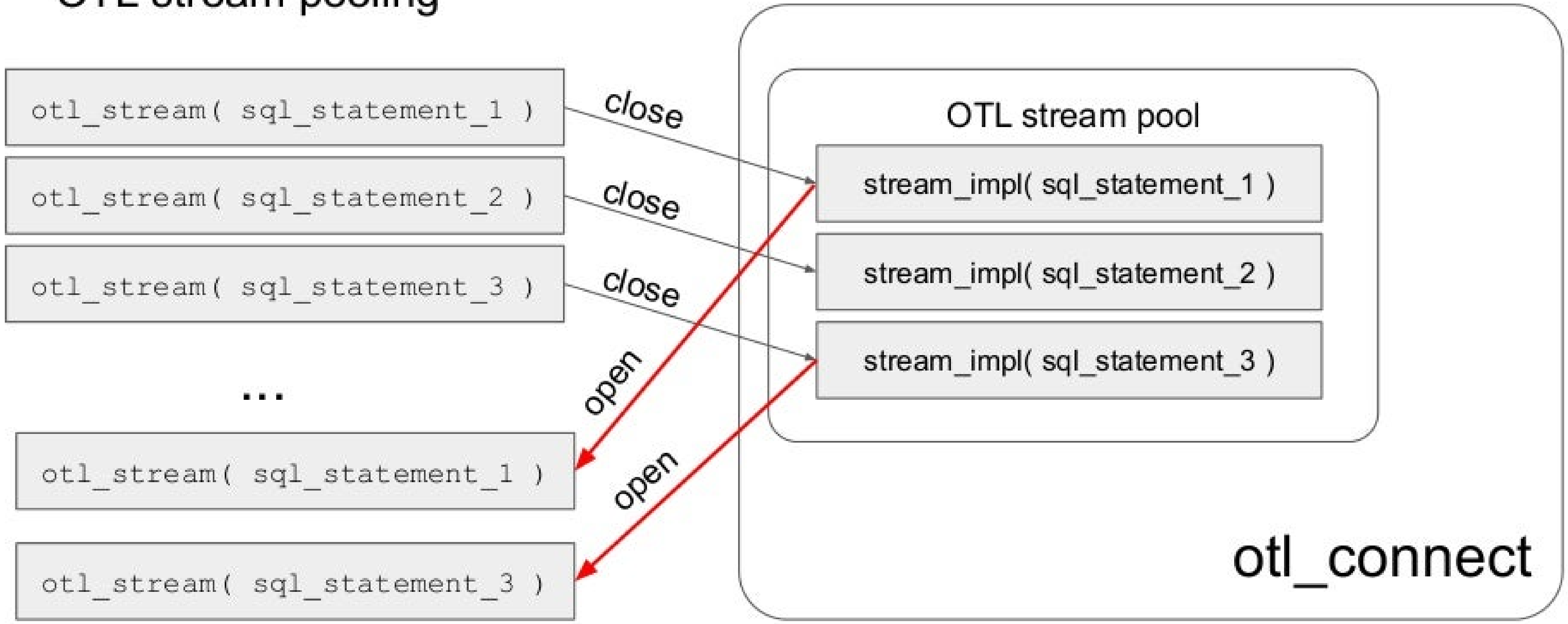

Application level
otl_stream
otl_exception
otl_long_string
...
otl_connect
OTL-adapters
OCI
SNAC
CLI
ODBC
Oracle
MS SQL Server
DB2
...
XXX

OTL

- Header-only
- Prepared statements;
- Transaction support;
- BLOB;
- Binding;
- Bulk IO.

OTL

OTL stream pooling


otl_stream(sql_statement_1)
otl_stream(sql_statement_2)
otl_stream(sql_statement_3)
close
close
close
OTL stream pool
stream_impl(sql_statement_1)
stream_impl(sql_statement_2)
stream_impl(sql_statement_3)
open
open
...
otl_stream(sql_statement_1)
otl_stream(sql_statement_3)
otl_connect

http://otl.sourceforge.net/otl3_stream_pooling.htm

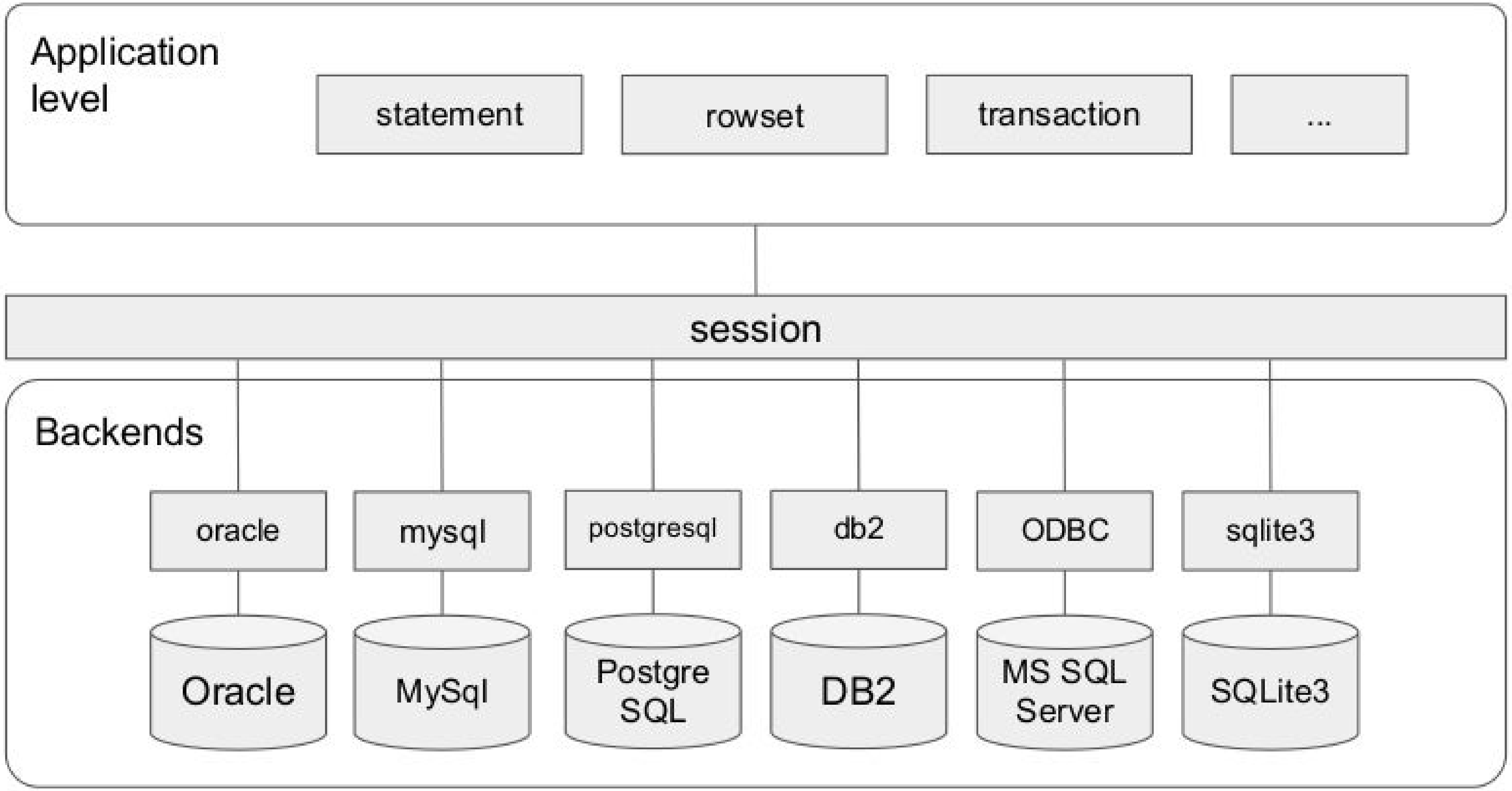
OTL

- Header-only
- Prepared statements;
- Transaction support;
- BLOB;
- Binding;
- Bulk IO.

OTL

```
#define OTL_ODBC_MSSQL_2008 // Compile OTL 4/ODBC, MS SQL 2008
#include <otlv4.h>
// ...
otl_connect::otl_initialize();
otl_connect db( "usr/pass@myserver" );

otl_stream selector(10, "SELECT id FROM employee WHERE name = :name<char[32]>" , db );
selector << "Albert";

std::vector< int > ids;
while( !select_query.eof() ) {
  int id;
  selector >> id;
  ids.push_back( id );
}

otl_nocommit_stream inserter(10, "INSERT INTO project (id, name, employeeid) "
                                 "VALUES (201, 'Manhattan Project', :id<int> )" , db );
for( auto id : ids ){
  inserter << id;
}

inserter.flush();
db.commit();
```

SOCI



Application level
statement
rowset
transaction
...
session
Backends
oracle
mysql
postgresql
db2
ODBC
sqlite3
Oracle
MySql
Postgre SQL
DB2
MS SQL Server
SQLite3

SOCI

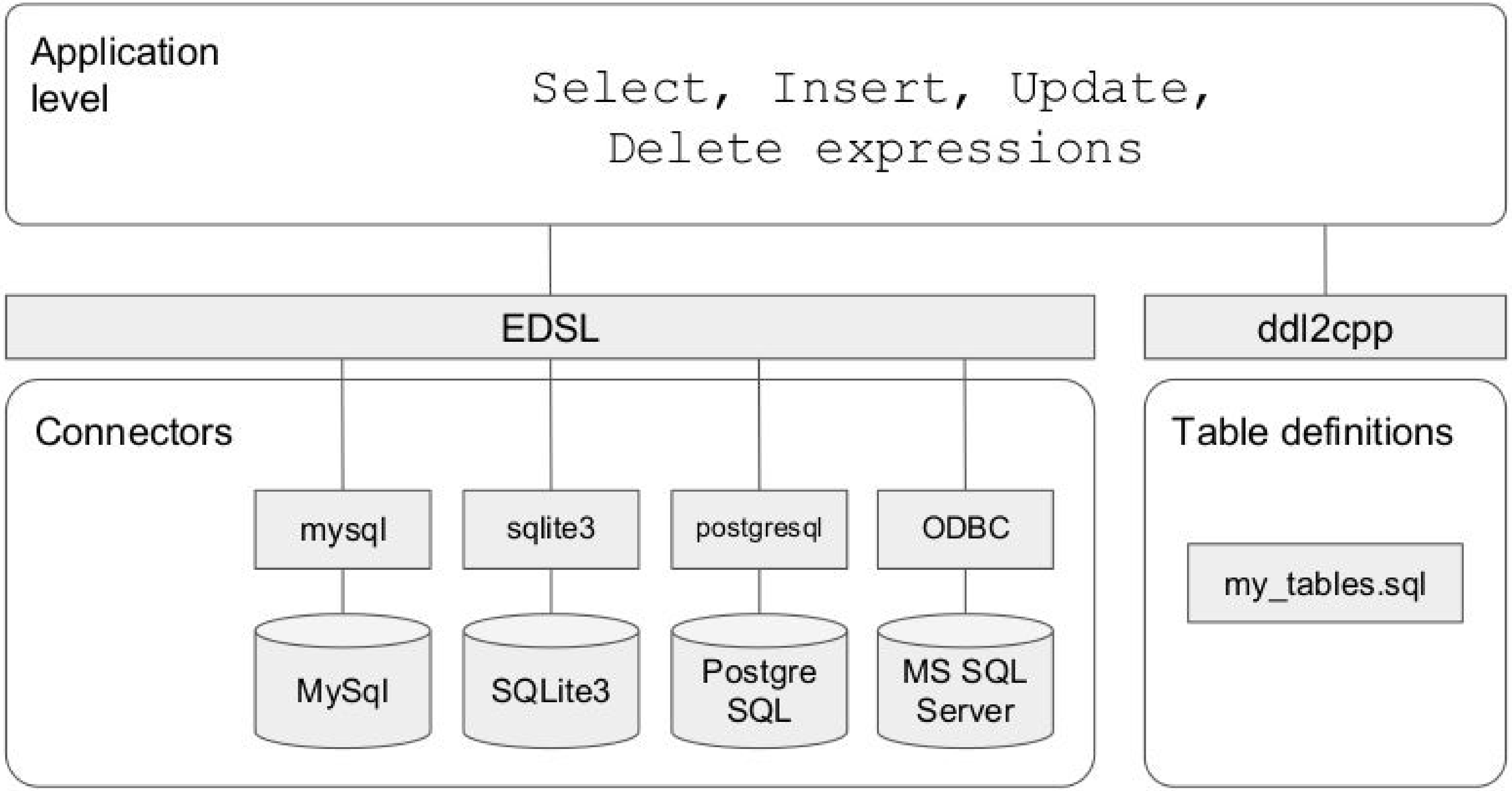
- Prepared statements;
- Transaction support;
- BLOB;
- Binding: named, positional;
- Bulk IO.

stiffstream

SOCI

```
soci::session ses( soci::postgresql, "dbname=db1 user=user password=123 ..." );
transaction trx( ses );
const std::string name{ "Albert"};
rowset< row > rs =
  ( ses.prepare << "SELECT id FROM employee WHERE name = :name", into(name) );

std::vector< int > ids;

for( const auto & row : rs ){
  ids.push_back( row.get< int >( 0 ) );
}

for( auto id : ids ){
  ses << "INSERT INTO project (id, name, employeeid) "
         "VALUES (201, 'Manhattan Project', :id )", use( id );
}
trx.commit();
```

Sqlpp11


Application level
Select, Insert, Update, Delete expressions
EDSL
ddl2cpp
Connectors
mysql
sqlite3
postgresql
ODBC
MySql
SQLite3
Postgre SQL
MS SQL Server
Table definitions
my_tables.sql

Sqlpp11

```
for( const auto& row :
       db( select( foo.name, foo.hasFun )
            .from( foo )
            .where( foo.id > 17  and
                    foo.name.like( "%bar%" ))))
{
    if (row.name.is_null())
         std::cerr << "name is null" << std::endl;
    else
    {
       // string-like fields are
       // implicitly convertible to string
       std::string name = row.name;

      // bool fields are implicitly convertible to bool
      bool hasFun = row.hasFun;
      do_something( name, hasFun );
    }
}
```

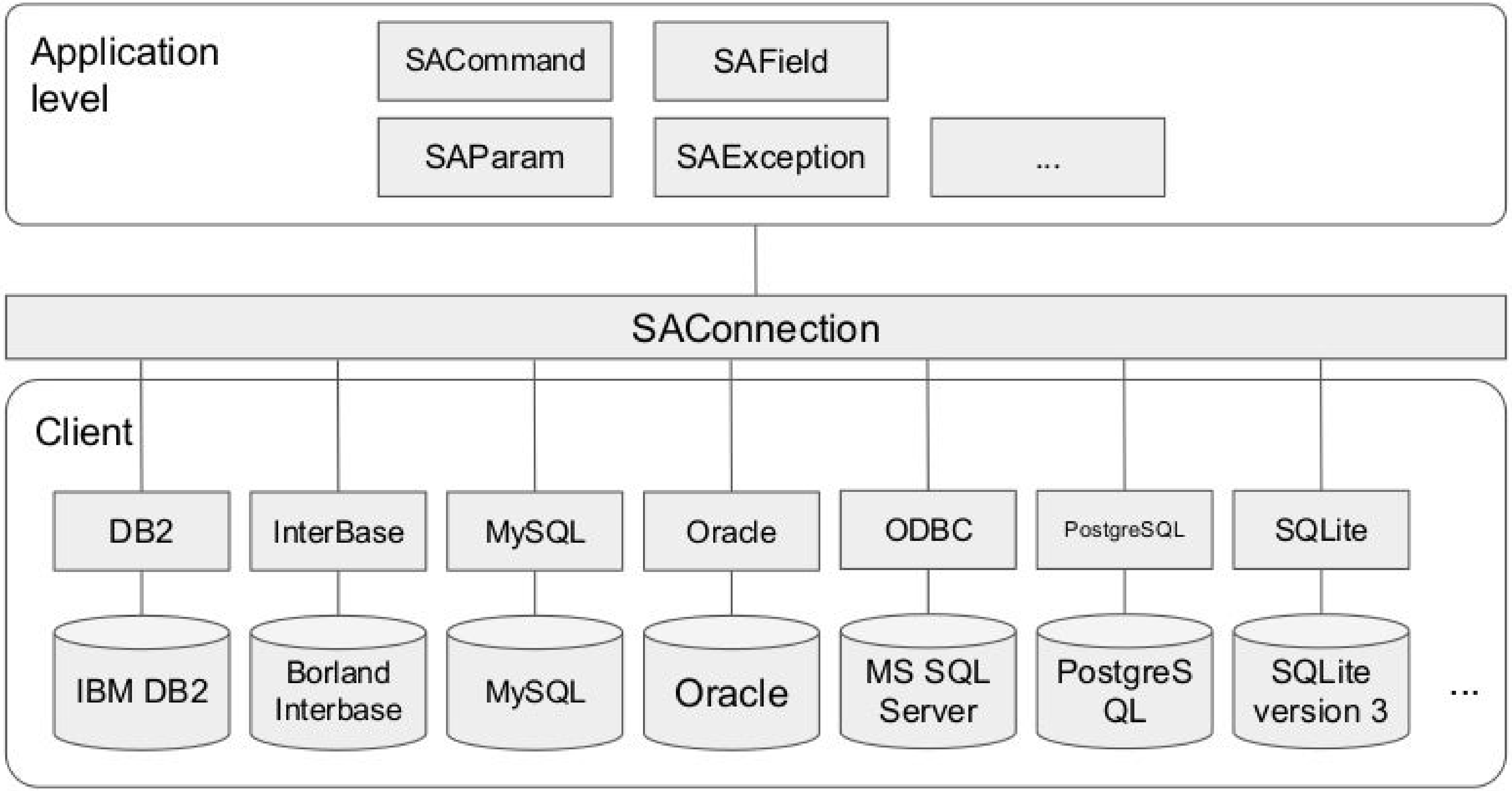
```
CREATE TABLE foo (
     id bigint,
     name varchar(50),
     hasFun bool
);
```

https://github.com/rbock/sqlpp11/wiki/Select

Sqlpp11

- Prepared statements;
- Transaction support;
- BLOB.

SQLAPI++


Application level
SACommand
SAField
SAParam
SAException
...
SAConnection
Client
DB2
InterBase
MySQL
Oracle
ODBC
PostgreSQL
SQLite
IBM DB2
Borland Interbase
MySQL
Oracle
MS SQL Server
PostgreS QL
SQLite version 3
...

SQLAPI++

- Prepared statements;
- Transaction support;
- BLOB;
- Binding;
- Bulk IO.

SQLAPI++

```cpp
SAConnection con;
con.Connect( "myserver", "user", "pass", SA_Oracle_Client);

SACommand cmd( &con, "SELECT id FROM employee WHERE name ='Albert'" );
cmd.Execute();

std::vector< int > ids;

while( cmd.FetchNext() ){
  ids.push_back( cmd[1].asLong() );
}

cmd.setCommandText( "INSERT INTO project (id, name, employeeid) "
                    "VALUES (201, 'Manhattan Project', ? )" )

for( auto id : ids ){
  cmd.Param(2).setAsLong() = id;
  cmd.Execute();
}

con.Commit();
```

Итого

	Prepared statements	Transaction	BLOB	Binding	Bulk IO	Complex Data types
QtSql	+	+	+	positional named	+	-
Poco::Data	+	+	+	positional	+	+
OTL	+*	+	+	positional	+	-
SOCI	+	+	+	positional named	+	-
sqlpp11	+	+	+	EDSL	-	-
SQLAPI++	+	+	+	positional named	+/-	-

Что будет завтра

cppstddb

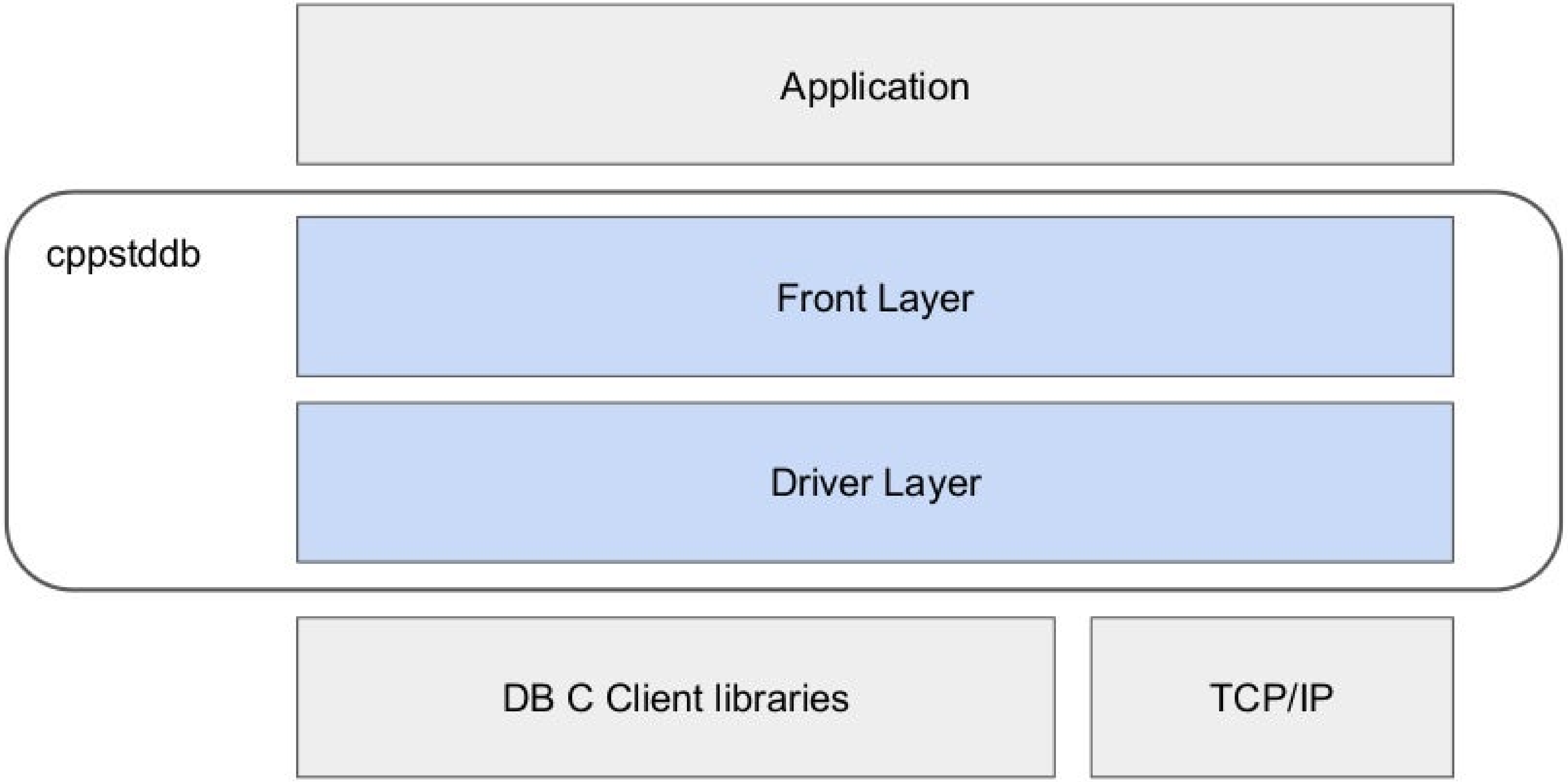
Презентация:

- https://www.youtube.com/watch?v=75aUjcYr6vE
- https://github.com/CppCon/CppCon2016/tree/master/Presentations/A%20modern%20database%20interface%20for%20C%2B%2B

Репозиторий:

- https://github.com/cruisercoder/cppstddb

cppstddb


Application
cppstddb
Front Layer
Driver Layer
DB C Client libraries
TCP/IP

cppstddb

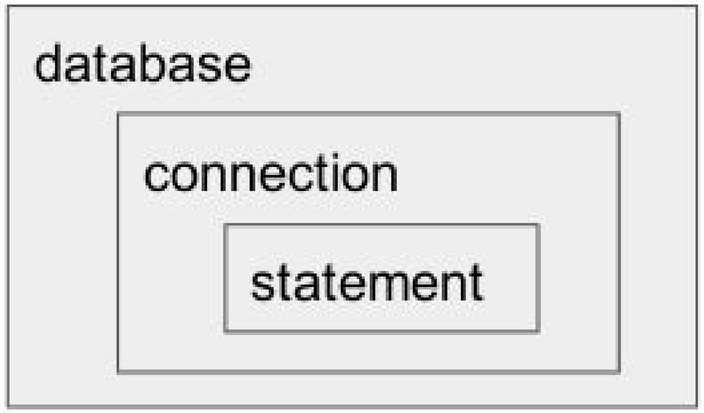
```
using namespace cppstddb::mysql;

auto db =
cppstddb::mysql::create_database();
db
    .statement(
        "select * from score")
    .query()
    .rows()
    .write(cout);
```

cppstddb

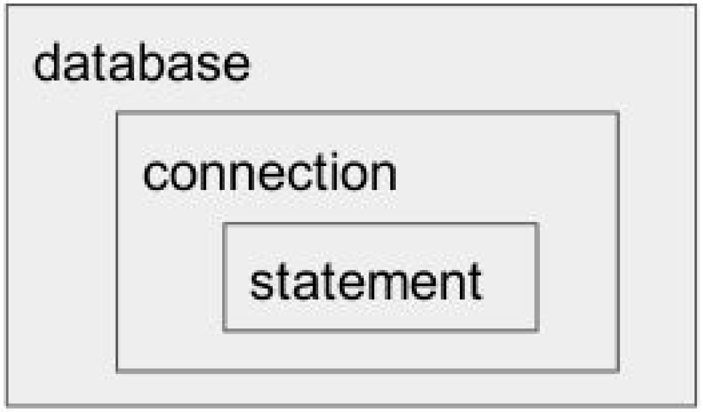
```
using namespace cppstddb::mysql;

auto db =
cppstddb::mysql::create_database();
db
    .statement(
         "select * from score")
    .query()
    .rows()
    .write(cout);
```

```
auto db =
cppstddb::mysql::create_database();
auto rowset =
  db.connection()
    .statement(
      "select * from score")
    .query()
    .rows();

for(auto row : rowset) {
  auto f = row[0];
  std::cout << f << "\n";
}
```

cppstddb

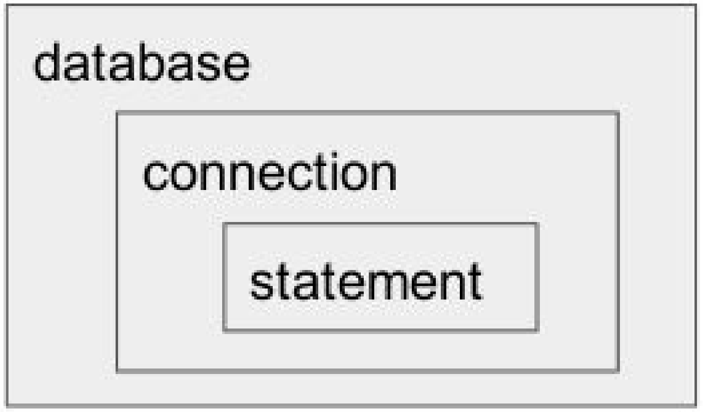


cppstddb



Id	Name	Age	Salary	WorksSince
1	Bob	31	3500$	2017-01-02
2	Jack	24	2500$	2015-05-04
3	Eric	33	3200$	2000-06-01
4	Alexander	62	500$	2015-11-06

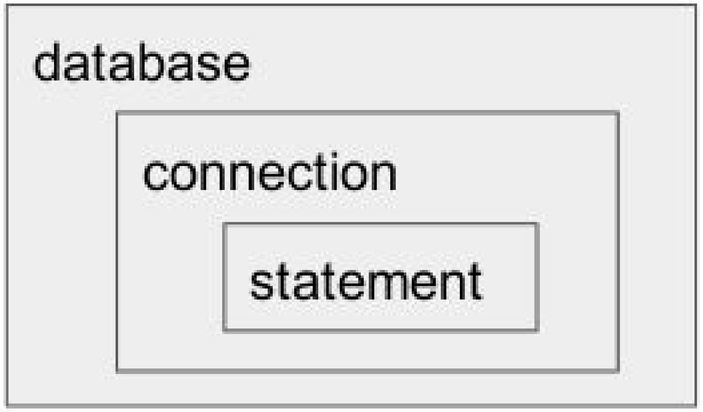
cppstddb



column

Id	Name	Age	Salary	WorksSince
1	Bob	31	3500$	2017-01-02
2	Jack	24	2500$	2015-05-04
3	Eric	33	3200$	2000-06-01
4	Alexander	62	500$	2015-11-06

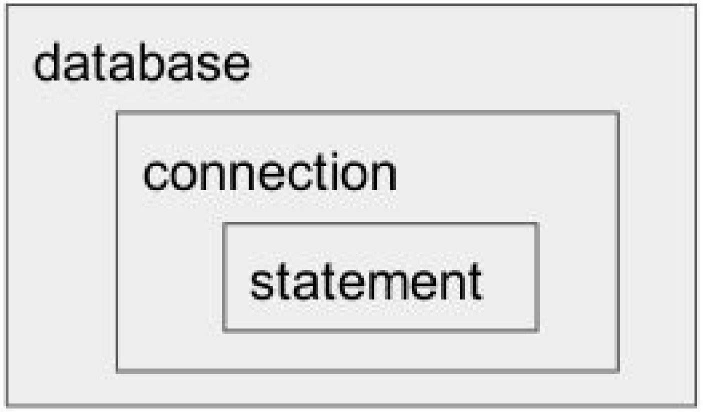
cppstddb

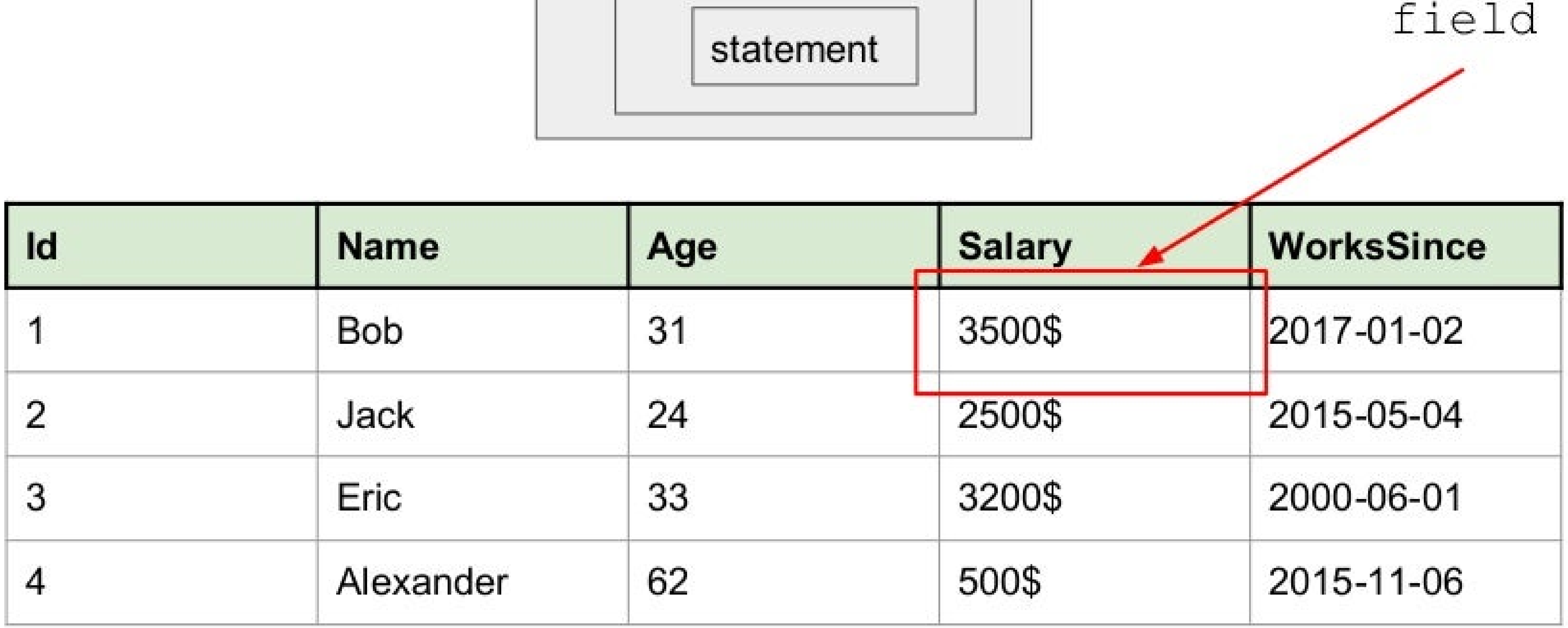


columnset

Id	Name	Age	Salary	WorksSince
1	Bob	31	3500$	2017-01-02
2	Jack	24	2500$	2015-05-04
3	Eric	33	3200$	2000-06-01
4	Alexander	62	500$	2015-11-06

cppstddb



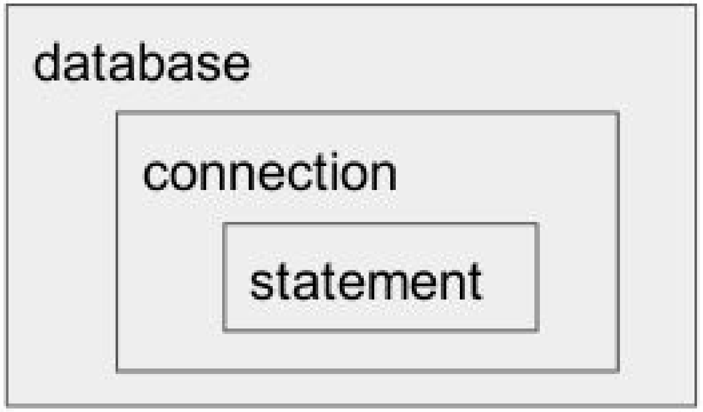


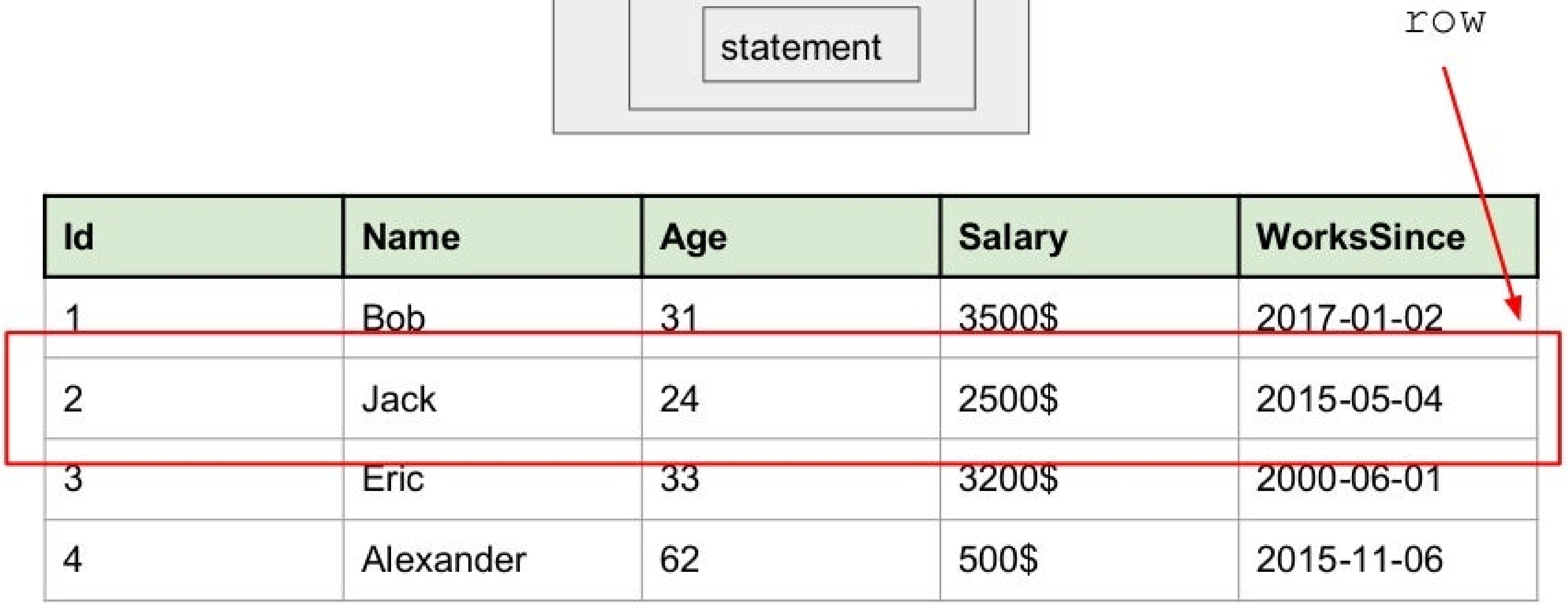
Id	Name	Age	Salary	WorksSince
1	Bob	31	3500$	2017-01-02
2	Jack	24	2500$	2015-05-04
3	Eric	33	3200$	2000-06-01
4	Alexander	62	500$	2015-11-06

cppstddb

```
for(auto row : db.query( "select name from students") ){

  auto name_field = row[ 0 ];

  // ok if not null
  auto name = name_field.as< std:: string >();

  // ok
  auto name_opt = name_field.optional< std:: string >();

  // error
  auto name_opt = name_field.as< int >();
}
```

cppstddb





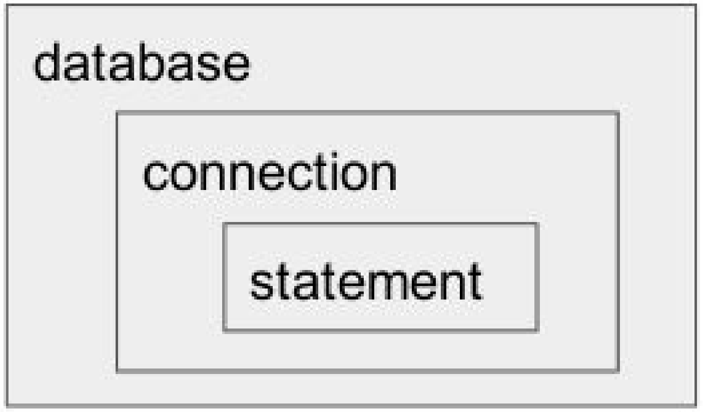
Id	Name	Age	Salary	WorksSince
1	Bob	31	3500$	2017-01-02
2	Jack	24	2500$	2015-05-04
3	Eric	33	3200$	2000-06-01
4	Alexander	62	500$	2015-11-06

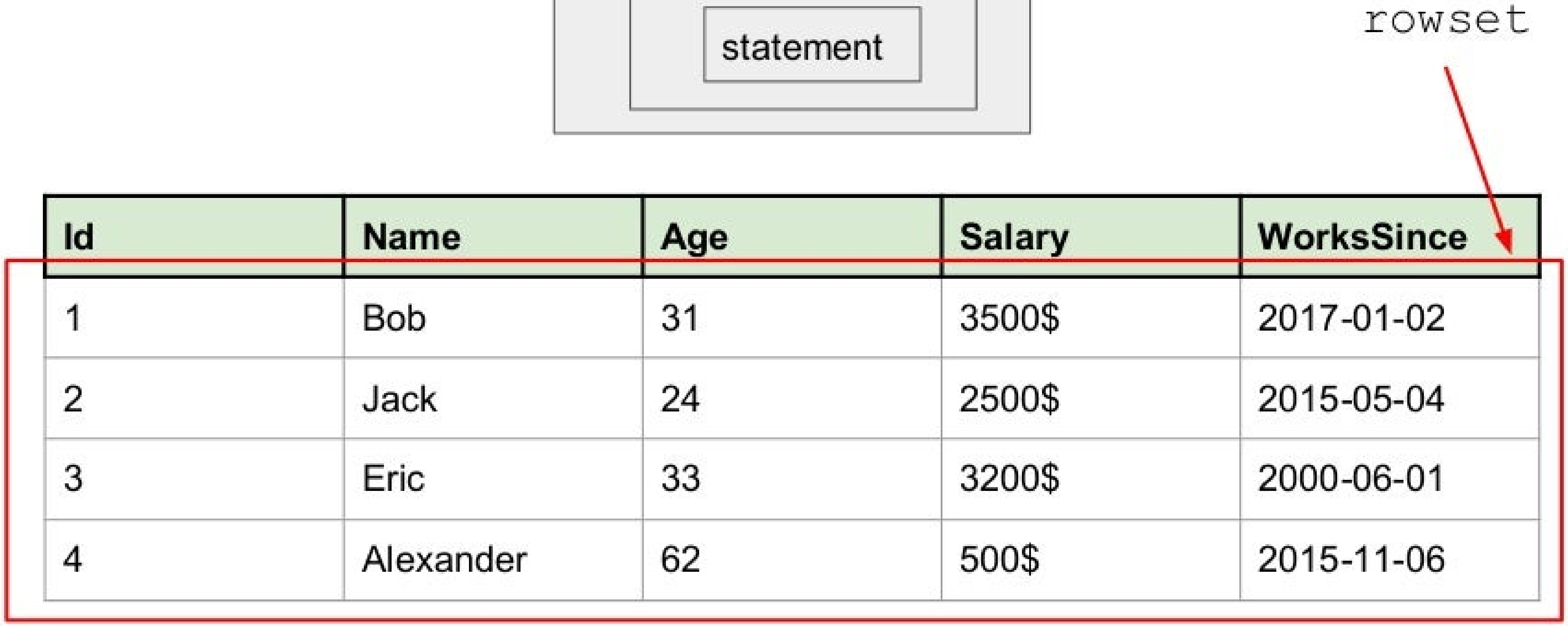
cppstddb

```
for(auto row :
      db.query(
       "select s, i, d from table") ){
  string s;
  int i;
  date d;

  row.into( s, i, d );
  // ...
}
```

```
auto field_s_idx = rows.columns()["s"];
auto field_i_idx = rows.columns()["i"];
auto field_d_idx = rows.columns()["d"];

for(auto row :
      db.query(
       "select s, i, d from table") ){
  string s = row[field_s_idx];
  int i = row[field_i_idx];
  date d = row[field_d_idx];
  // ...
}
```

cppstddb





Id	Name	Age	Salary	WorksSince
1	Bob	31	3500$	2017-01-02
2	Jack	24	2500$	2015-05-04
3	Eric	33	3200$	2000-06-01
4	Alexander	62	500$	2015-11-06

stiffstream

cppstddb

```
using namespace cppstddb::mysql;
auto db = create_database("mysql://server/db");
auto con = db.connection();
auto stmt = con.query("select * from person");
auto rows = stmt.rows();

for (auto i = rows.begin(); i != rows.end(); ++i) {
  for(int c = 0; c != row.width(); ++c) {
    auto field = row[c];
    cout << "value: " << field << "\n";
  }
  cout << "\n";
}
```

cppstddb

```
for(auto row :
      db.query(
       "select s, i, d from table") ){
  string s;
  int i;
  date d;

  row.into( s, i, d );
  // ...
}
```

```
for(auto row :
      db
        .row_array_size( 100 )
        .query(
       "select s, i, d from table") ){
  string s;
  int i;
  date d;

  row.into( s, i, d );
  // ...
}
```

stiffstream

cppstddb

```
auto insertion_stmt =
  db.statement( "INSERT INTO employees( name, date, salary ) "
                "VALUES( ?, ?, ? ) " )
    .row_array_size( 100 );

for( auto em : new_employees ){
  insertion_stmt.query( em.name(), em.date(), em.salary() );
}
```

cppstddb

Что еще в планах:

- поддержка “всех” СУБД;
- сериализация объектов;
- отвязанные от соединения датасеты;
- неблокирующие операции.

stiffstream

Вопросы?

Николай Гродзицкий
ngrodzitski@stiffstream.com