数据库接口在总体的层次上分为二层。

分别是DAO层，Interface层。

其中DAO层，封装所有应用程序对数据库的逻辑操作。比如实时数据库的测试连接、读取历史数据、写入历史数据。比如关系数据库的 测试连接、创建数据表、写入或者更新数据、读取数据。

Interface层，就是多数据库接口，相当于DAO的数据库驱动层。用于提供DAO标准接口操作到不同底层数据库的映射。该功能类似于SSH下的hiberate的功能。

实时数据库RTDBInterface

实时数据库RTDBInterface用来实现实时数据库连接对象的创建、使用和销毁等管理功能。RTDBInterface主要分为两层，即对外提供实体连接的RTDBHelper层，和对应数据库的Helper层。两层之间由工厂类负责创建联系。

使用工厂类，根据配置文件对外提供短连接接口RTDBHelper和长连接接口RTDBHelperLong。

其中短连接接口，就是在RTDBHelper中的每一个方法，在执行结束后，均要断开连接。如果数据库支持，则短连接本质上使用的是连接池技术。如果数据库不支持连接池技术，则短连接技术由于每次耗时较长，则不适合用于频繁读写的场合。仅用于读写频率比较低的场合。

**特别注意上面的描述，即使不支持线程池的的实时数据库，短连接也是需要的，在读写不平凡的场合需要使用短连接。**

长连接接口，就是在RTDBHelper中的每一个方法，在执行结束后，均不断开连接。并且每个方法在执行前，均检查连接是否有效，如果无效，则先重新建立连接，在进行方法的操作。

PIHelper

GoldenHelper

PGIMHelper

DBInterface.RTDBInterface

RTDBHelper

DBInterface.RTDBInterface.LongXXX

RTDBHelperLong

BaseRTDBHelper

RTDbHelperFactory

对于对应数据库的Helper层，要保证这些Helper层无论应对RTDBHelper还是RTDBHelperLong，都是一样的。

**也就是无论长连接还是短连接，Helper层要保持一致。要完全通过RTDBHeper层的差异，来实现短连接和长连接。**

**RTDBHeper层在实现了长连接和短连接的差异后，还要保证，从DAO调用时的模式是完全一致的。也就是说，仅需要通过转换RTDBHeper和RTDBHeperLong既可以完成长连接和短连接方法的转换，而无需修改DAO任何代码。**

PIHelper分析。

**PIHelper**

ConnectionString{Set}

通过连接字符串设定内部参数：

—\_server

—\_port

—\_userId

—\_password

如果输入为空，则保持原有默认值。

再用\_userId和\_password构成PI真正需要的连接字符串\_connectionString

\_connectionString