XX数据仓库建设规范

版本号V1.0

修订历史

版本号	作者	内容提要	核准人	发布日期
1.0		初稿		

1 概述

本文档制定了XX数据仓库中数据库对象的命名规范（用户、表、视图、存储过程、函数、表分区、主键、索引、序列等）、数据库编程规范，JAVA编程规范为系统设计和开发工作提供统一的命名标准，提高系统的规整性和代码的可读性，减轻维护工作量，提高工作效率。

2 数据库对象命名规范

2.1 层次划分

序号	模型层次	用途
1	ODS	存放来自各个系统的原始数据；
2	DW	根据业务分析需求，对主题域内的数据进行轻度汇总；
3	DM	建立跨域的业务主题模型，比如中高端用户，拍照用户等；不能进行同层引用；
4	DIM	统一服务于数据中心的参数表；
5	APP	应用层，用于生成报表
6	XX	XX

数据层级按照自己数据仓库规划的命名即可~

2.2 表、视图、存储过程、函数命名规范

<对象类型><_模型层次><_主题><_对象描述>[_汇总类型][_存储类型]

说明：<> 尖括号中的内容为必须项，适用于所有用户层对象

[] 方括号中的内容为可选项，会因用户层及对象的不同而不同

命名约束：数据库对象命名可能受最大长度限制,因此在实际命名中如果按照规范约定的命名方式存在超长的现象,需要开发人员灵活控制。

2.2.1对象类型

<对象类型><_模型层次><_主题域><_对象描述>[_汇总类型][_存储类型]。

适用范围：所有用户层对象。

对象类型	对象	说明
TB	TABLE	表
VW	VIEW	视图
……	……	……

2.2.2模型层次

<对象类型><_模型层次><_主题域><_对象描述>[_汇总类型][_存储类型]

说明：对象属性一般为对象归属用户的简写。

适用范围：所有用户层对象。可以参照自己的对象属性命名规范，对此不要求统一。

模型层次	说明
ODS	获取层，存放从各个源系统接收的原始数据；
DW	根据业务分析需求，对数据进行汇总；
DM	建立跨域的业务主题模型；
DIM	维表
APP	报表层，根据DM模型数据生成报表。

2.2.3主题域

<对象类型><_模型层次><_主题域><_对象描述>[_汇总类型][_存储类型][_][序号或描述]

说明：主题域是对数据进行大类划分，不同用户下的分类有所不同。

适用所有业务层；每个新增的业务主题均需到该规范备案登记。

主题域	命名	简称	描述
客户域	Customer	XX	泛客户
…	…	…	…

2.2.4对象描述

<对象类型><_模型层次><_主题域><_对象描述>[_汇总类型][_存储类型]

- 适用范围：所有用户层对象；
- 对象描述要求简洁准确，尽可能的直观表达对象的含义，通常包含业务+功能；

如果是通用命名规则：<对象类型><_模型层次><_主题域><_对象描述>[_汇总类型][_存储类型]，这里的对象描述是多业务的合成体，这时不加业务。

- **汇总类型**

<对象类型><_模型层次><_主题域><_对象描述>[_汇总类型][_存储类型]

适用范围：除字典表、日志表之外的对象。

描述	汇总类型
日	DAY
月	MON
年	YEAR

2.2.5存储类型

<对象类型><_模型层次><_主题域><_对象描述>[_汇总类型][_存储类型]

适用范围：所有用户层除日志、字典表、维表之外的对象。

对象描述	存储类型	说明
目标程序	无	
临时表	TMP	程序中临时使用的中间表，用于存放程序运行中使用的临时数据，程序运行结束后表由程序自行清空，只保留结构。
配置表	CFG	

2.2.5.1 日表

日表以统计周期字段做日分区。数据保留周期为业务需要的周期，月底最后一天的数据不保存，如有需要则沉淀到月表中。

2.2.5.2 月表

月表以统计周期字段做月分区。除该字段外，其余字段与日表必须相同。

数据保留周期为业务需要的周期。 所有的月报表、月KPI数据必须从月表出，禁止从日表出。

2.2.5.3 周表

周表数据保留周期为业务需要的周期。

2.3 其他对象命名规范

对象	命名规则	说明

表分区	根据实际情况自行确定	建议等
主键	PK<_表名><_列名>	
索引	IDX<_表名><_列名>	
…	…	…

2.4 常用字段命名规范

字段名	数据类型	字段说明	备注
常用字段1	常用类型1	字段说明1	备注1

字段命名需做到见名知其意，避免用中文拼音，或者拼音+英语的方式。

可以参考企业现有业务数据库的数据字典命名。

2.5 常用单位规范

约定数据仓库中字段的默认单位，比如车速默认单位是KM/h。

2.6 数据库对象命名注意事项

- 命名尽量采用富有意义的英文词汇，不准采用汉语拼音。
- XX……

2.7 数据仓库建表注意事项

- 表名，列名等需要添加注释，否则不予上线。
- XX……。

3 主机目录及文件命名规范

3.1 用户命名规范

主机用户名命名规范：

序号	主机用户名	账号类型	用途
1	hadoop	应用程序账号	hadoop集群管理用户
2	…	ftp账号	…

3.2 目录规划

<根目录>/<二级目录>/[三级目录/]<业务域>[/自定义]

目录规划不做强制性的要求，但是要做到层次清晰、命名规范，见名知意。

- 根目录

  取值为：根据物理存储挂载情况而定；

- 二级目录

  取值为：主机如果没有文件系统挂载点，则二级目录为用户家目录，否则取值用户名；

- 三级目录

  取值为：用户自行定义，如果存储在用户家目录下则需要三级目录；

- 业务域

  取值为：抽取的文件按业务类型进行分类存储。

业务简称	说明
业务1	说明1

业务2	说明2

- 自定义

  取值为：可选项，如果文件存储有其它要求，可根据实际情况灵活调整,如需要分省存放等。

3.3 文件命名规范

<文件类型>_<主题域>_<数据周期>_<接口文件序号>.dat

- 主题域

  主题域取值情况咱定为各项目名称：

- 数据周期

  取值为:周期日数据8位长度,YYYYMMDD,月数据6位长度YYYYMM;

- 接口文件序号

  取值为：接口文件序号长度为3，默认从000开始；

3.4 文件格式规范

- 文件分隔符

  文件字段尽量不采用定长分隔，采用“|”等特殊字符作为分隔符,另外在抽取文件时需要确定字段内容中不会出现分隔符字符，以免错列；

- 文件编码

  文件编码采用UTF-8。

4 数据保存周期规范

周期类型	模型层次	保留周期(HIVE)	备注
日	ODS	365	……

5 数据库编程规范

5.1 参数和变量命名规范

5.1.1对象变量

对象变量命名规则如下：

- 命名规则：<对象类型><_><变量描述>
- ……

5.1.2参数和对象命名注意事项

- 所有名称采用英文单数名词或动词，避免出现复数。
- 固定长度的字符串类型采用char,长度不固定的字符串采用varchar,一定要避免长度不固定的情况下采用char。
- 如无特殊需求，避免使用大字段(blob, clob, long, text, image 等)。
- 命名使用英文单词，避免使用拼音，特别不应该使用拼音简写。命名不允许使用中文或者特殊字符。
- 命名中若使用特殊约定或缩写，则必须要注释说明。
- 使用有意义、易于记忆、描述性强、简短及唯一的英文单词。自已特有的命名风格，要自始自终保持一至，不可来回变化。
- 对于变量命名，禁止取单个字符(如i、j…)，建议除了要有具体含义外，还能表明变量类型等。
- 除非必要，不允许使用数字或较奇怪的字符来定义标识符。

5.2 书写规范

5.2.1代码大小写规范

- 所有数据代码统一使用小写字母书写，以方便不同数据库之间的移植，同时也避免程序调用问题。参数和局部变量，全局变量用大写。

5.2.2代码缩进规范

- 程序块采用缩进风格书写，保证代码清晰易读，风格一致。缩进格数统一为 4 格。
- 必须使用空格,禁止使用TAB键。
- 同一条语句占用多于一行时,每行的第一个关键字应当右对齐。
- 对于Insert … values 和update 语句，一行写一个字段，这段后面紧跟注释（注释语句左对齐），values 和insert 左对齐，左括号和右括号与insert、values 左对齐。
- insert...select语句中，应使每行的字段顺序对应，以每行不超过80字符为准，以增强可读性。

5.2.3空格及换行

- 关键字之后要留空格。
- 创建表、存储过程、函数时，表名、存储过程名和函数名之后不要留空格。
- 不允许把多个语句写在一行中，即一行只写一条语句。
- 相对独立的程序块之间、变量说明之后必须加空行。
- 超过80字符的语句要分行书写，长表达式应在低优先级操作符处换行，操作符或关键字放在新行之首。划分出的新行应适当地缩进，使排版整齐，语句可读。
- if后的条件要用括号括起来，括号内每行最多两个条件。
- 不同类型的操作符混合使用时，建议使用括号进行隔离，以使代码清晰。
- 减少控制语句的检查次数，如在else(if…else)控制语句中，对符合条件频率高的尽量放到前面。
- 尽量避免使用嵌套的if语句，在这种情况应使用多个if语句来判断其可能。

5.2.4其它

- 避免使用select * 语句。
- insert 语句必须给出字段列表，否则对后续表的扩展回带来维护上的麻烦。
- 当一个SQL 语句中涉及到多个表时，必须使用别名来限定字段名，这使其它人阅读起来更方便，避免了含议模糊的引用，其中能够别名中清晰地判断出表名。
- 确保变量和参数在类型和长度与表数据列类型和长度相匹配。

5.3 注释规范

- 一般情况下，源程序有效注释量不低于30%以上。

说明：注释的原则是有助于程序阅读理解，便于后期维护，在该加的地方都加了，注释不宜太多但也不能太少，注释语言须准确、易懂、简洁。

- 所有变量定义需要加注释，说明该变量的用途和含义。
- 注释内容要清晰明了，含义准确，防止注释二义性。
- 禁止在注释中使用缩写，特别是非常用的缩写。
- 注释与所描述代码进行同样的缩排。
- 对程序分支必须书写注释。
- 保证代码和注释的一致性。修改代码同时修改相应的注释,不再有用的注释要同步删除。
- 注释应与其描述的代码相似,对代码注释应放在其上方或右方(单条语句的注释)相应的位置,不可放在下面。
- 注释上面的代码应空行隔开。
- 统一文件头的注释。
- 在代码的功能、意图层次上进行注释，提供有用、额外的信息。
- 函数应对返回代码详细描述。
- 尽量使用”#”进行注释。
- 避免在一行代码或表达式的中间插入注释。
- 所有硬编码必须加注释，如 id='0' 则需要优先注释 '0'的含义， 或者在注释中说明对应的字典表。

5.4 语法规范

- **所有DDL和DML语句尽量遵循标准SQL，以SQL99为基准。**

说明：采用标准SQL编写，方便移植时各种数据库之间做对应修改。

正例：

```
delete from table1;
```

反例：

```
delete table1;
```

- **数据类型采用基本数据类型，尽量不要使用某数据库特有的类型。**

说明：采用基本数据类型，各种数据库均支持，减少不同版本的维护。设计数据类型和长度时要考虑应用编程开发的方便以及后续可维护性。

- 对于特别复杂的sql(特别是多层嵌套,带字句或相关的查询),应先考虑是否设计不当引起,对复杂的sql可以通过程序实现,原则上遵循一句话只做一件事情,避免多重嵌套SQL的使用。必要时采用中间表。
- 对于超过2个以上的大表关联，必须进行执行计划验证，并在设计中有所体现。
- 避免隐式的数据类型转换。

- 不要将空的变量值直接与比较运算符比较。如果变量可能为空,应该使用is null或is not null来进行比较。
- 每个程序过程生成的目标数据表不允许出现空值。
- 尽可能地使用相关表字段的类型定义,如%type,%rowtype等。
- 存储过程中变量不允许在代码中随意定义变量。定义变量时，完成相同功能的变量尽量放在一起，不同功能逻辑的变量用空行隔开，以增加代码的可读性。
- 对数据库脚本代码中所定义的变量要进行初始化。
- 复合主键的字段数不超过4个。
- **……**

5.5 程序结构

5.5.1 程序结构模板

5.5.2 程序代码

5.5.2.1 程序代码规范

- 程序代码段左侧要留有 1 个缩进（4 个空格）。
- 除特殊程序(如空调度、日志程序等)外，程序开始、程序结束、程序出错时都要记录日志，日志记录使用公用的函数或存储过程，具体使用方法参见后面日志内容。
- 关键字要换行输写，不同行关键字要右对齐。
  - 关键字：insert、select、from、where、and、order(by)、group（by）、having、update 等。
- 对于内容超过一行的代码，换行时要有一个缩进，并注意对齐以保证美观。
- insert、select、group by、set 等涉及多个字段的操作，要分行列出每个字段，后续字段要有一个缩进，缩进字段要左对齐。
- 字段分隔符“,”要放在字段前面。
- insert 的字段要使用“()”括起来，括号独站一行，与 insert 对齐，括号内的字段要与括号有一个缩进。
  - 每个字段后面都要有字段说明(字段描述、值内容、单位等)，字段说明要对齐。
  - 字段说明内容可以换行，但同样要与上行字段说明内容对齐。
  - 对于比较简单的 SQL 语句，也可根据实际情况写在一行或几行中，但多行的要注意缩进，并且要注意美观性。
- 对于 insert 字段数量比较多的语句，对应的 select 中的字段尽可能定义别名，别名要与 insert 中字段名相同，这样很容易找到字段的对应关系。
- from 的表名要分行列出。
  - 嵌套 SQL 要使用“()”括起来，括号独站一行，与上一级语句要有一个缩进；括号内的 SQL 与括号还要有一个缩进的开始。
  - 对于多层嵌套，一定要注意各层嵌套的缩进层次，才能保证代码良好的可读性，否

则代码将非常难读。

- 程序中能不用动态 SQL 就不要用，因为立即执行 SQL 方式在程序编译时即可发现问题，而动态 SQL 只能执行时才能发现问题；这样，SQL 语句中处理分区表数据时，不要使用分区名方式，应使用分区字段做条件字段，mysql 会自动识别分区字段来处理分区数据。
- 关键字、保留字之间必须留有空格。
  - 运算符(赋值运算会、条件运算符、算术运算符等)前后各要留有一个空格。
  - 调用函数或存储过程时，传入的参数之间要留有一个空格(“,”之后，后续参数之前)。
  - 对于不分行的字段之间也要留有一个空格(“,”之后，后续字段之前)。
- 程序代码中的其它注释内容，也要注意对齐情况，位置视情况而定。
- 程序中所有涉及的表、视图、序列、存储过程、函数等，都必须加上用户名。
- 程序中不同用途的各段代码上方，要加上各自的语句标注，每段语句标注与上方的代码之间要留有一个空行。
- ……

5.5.2.2 程序代码示例

5.5.3程序日志

5.5.3.1 日志分类

程序日志分为两种：

- 一种是记录程序运行状态情况，一个程序运行一次只记录一条日志，包括程序名称、目标表名、统计时间、程序运行开始和结束时间、运行状态、出错位置和出错信息等，用于简单查询程序运行情况，以及以后可能的日志监控。
- 一种是记录程序运行过程情况，一次程序运行会记录多条日志，每条日志记录程序中不同阶段的运行状况，用于跟踪程序中各阶段的运行状况。与单条日志记录相比，时间上只记录运行开始时间即可。
- ……

5.5.3.2 日志记录

5.6 分区管理规范

- 分区表的分区增加、分区删除操作，统一由分区控制程序完成，应用数据处理程序中不允许包含增加、删除分区的操作；分区表清空分区的操作，应在应用数据处理程序中进行，这样可以避免因为程序多次运行导致的数据重复。
- 保留多个周期数据的表必须建立分区,分区键可以根据业务需要和数据大小分为日、月、年，这样即可以避免因为表越来越大导致程序运行速度越来越慢，又解决分区太多浪费空间。全量替换的数据表（如维表、临时表）可以不建立分区。
- 日分区表禁止保留月底最后一天数据，如果要用到月底最后一天数据，需要单独建立月表保存。

6 JAVA 编码规范

6.1 避免引发错误的编写规范

- 使用字符串的equals方法比较判断时，如有常量字符串，一定要养成常量在前，变量在后的编写习惯。
- 养成这种编码习惯能够有效减少当比较的变量是null时发生空指针的错误
- 在finally中执行关闭操作,能够确保出现异常时数据库连接、IO读写句柄被正常关闭。
- ……

6.2 编程注意事项说明

- 明确方法功能，精确（不是近似）地实现方法的设计。一个方法仅完成一件功能，即时简单功能也应该编写方法实现。
- 异常捕获后，如果不对改异常进行处理，则应该记录日志或使用。
- 如果是自己抛出的异常，则必须要填写详细的异常描述信息，这样才能方便。
- 判断时要注意运算符的优先级，并用括号明确表达式的操作顺序，避免使用默认的优先级。
- 不要在循环体内定义变量。

- ……

6.3 程序排版及注释规范

6.3.1程序排版

- 程序块要采用缩进风格编写，缩进的空格数为4个。
- ……

6.3.2程序注释

- 一般情况下，源程序的有效注释量必须在30%以上。
- 类或者接口的注释需要包含其功能描述、使用方法、注意事项、作者、始于那个版本。
- ……

7 shell 编码规范

7.1 shell 编程案例