项目：（Sakila数据库）

项目简介：

- Sakila 的主要目的是支撑 DVD 租赁商店的业务流程，下面列举了一些业务流程活动中的关键点来理解数据库是如何支撑的（2006数据）
- 每个商店维护自己的租赁影片清单，当客户取走或归还DVD 光盘时会有一个专门的店员对这个清单进行维
- 护影片描述的内容同样在维护信息范围之内，如分类（动作、冒险、喜剧等）、演员、等级、特殊分类（例如被删除的情节和预告片）。这些信息可能被打印在DVD 包装的标签上
- 必须在商店注册成为会员才可以租赁光盘
- 客户可以在任何一家商店租赁一张或多张光盘，同时，商店希望客户在每张光盘对应的租赁期内归还之前租赁的光盘
- 顾客可以在任意时间对任何租赁的光盘付费

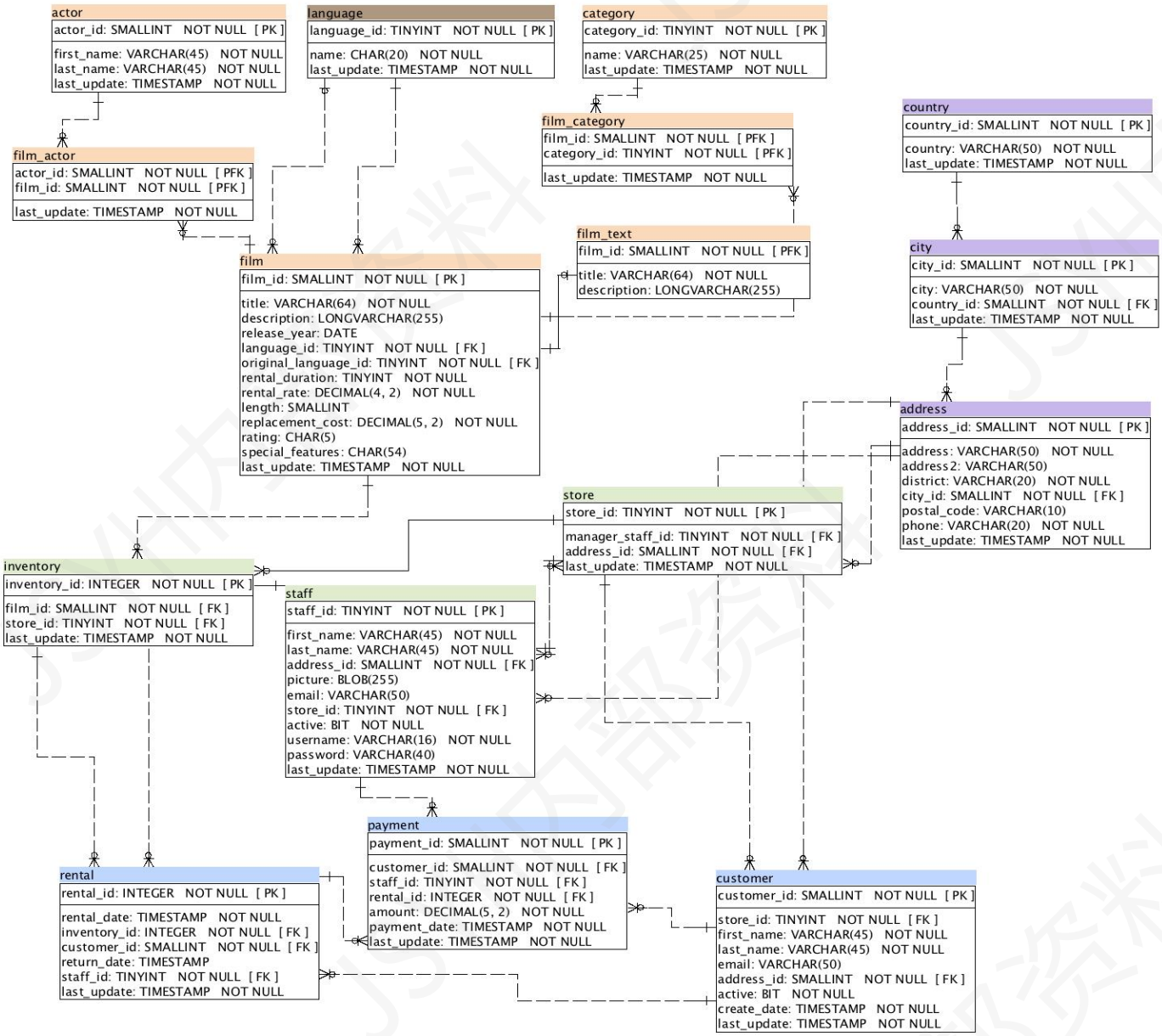
项目需求：

- 描述：实现系统定期从原数据库saklia抽取增量数据，然后转换成符合星型模型的数据，最后把数据加载到目标数据库的**租赁业务星型模型**中（其模型中包含2000-2010年间的所有数据）。

Sakila 数据库的表：

序号	表名(英文)	表名(中文)
1	actor	演员表
2	address	地址表
3	category	类别表
4	city	城市表
5	country	国家表
6	customer	客户表
7	film	电影表
8	film_actor	电影_演员表
9	film_category	电影_类别表
10	film_text	电影_文本表
11	inventory	库存表
12	language	语言表
13	payment	付款表
14	rental	租赁表
15	staff	工作人员表
16	store	商店表

Sakila 数据库的模型关系图:



Sakila 数据库的总体设计规范：

- 主题分类：
  电影类（黄色）：包含film表和包含影片附加信息的表，如category、actor和language。
  商店类（绿色）：包括store表和相关联的staff表、inventory表。
  客户类（蓝色）：以customer表为主线，包含与顾客有关联的rental表和payment表。
  区域类（紫色）：包括country、city和address表，这些表为顾客、商店和员工提供标准化的字典信息。
- sakila数据库表采用单一对象名称命名。
- 每张表都有自增主键列，列名采用“ 表名_id ”的规则命名，如表film的自增主键列为film_id。
- 外键约束引用主键，且名字与主键列相同。例如，表store的address_id列引用表address的address_id列。
- 每张表都有一列叫做last_update，这是一个TIMESTAMP类型的字段，用来记录增加或更新数据时的时间。

租赁业务的星型模型：

- 如图展示了一个典型的维度模型，它包含了一个叫做fact_rental的事实表，事实表与多个维度表关联。fact_rental表与sakila模式下的原始表rental表对应：rental表中的一行生成fact_rental表中的一行。
- 这种维度建模方式非常适用于联机事物处理（OLTP）。它同样也是一个经典的星型模式，因为几乎所有的维度都是单一的，维度表之间没有关联，维度表只和事实表有关。
- 注意：并不是所有的维度表之间都没有关联，比如图中的dim_ator和dim_film之间通过中间表dim_film_actor_bridge进行关联。

事实表说明：

- 事实表包含了一些数值类型的能体现业绩的业务度量值（count_returns、coun_rentals、rental_duration）。此外，还包含了一些列，用来作为指向维度表的键。当用户访问某个度量值时，维度表中的数据将提供该度量值对应的业务维度。

维度表说明：

- 租赁业务的星型模型中提到租赁业务星型模型的每一个维度都是一个单独的维度表。星型模型维度表的命名全部遵守了 dim_<dimension-name>的规则，其中 <dimension-name> 用来描述维度的内容。根据之前sakila数据库模型的分类，（建议）可以将维度表按照相关概念分为四组，再加上一组事实表“ how much？”，一共五组。

- 人员（who）：这组包括dim_customer表和dim_staff表，分别代表租赁业务中的客户和员工。维度表里使用 %_version_number、%_valid_from和%_valid_through列来跟着同一客户或员工的历史记录。

- 时间（when）：这组中的维度表主要用来记录所有光盘租赁或归还时间点，其中，维度表dim_date实际是日历，它所谓的角色扮演维度，用来同时标记租赁日期和归还日期。而dim_time维度表则是用来记录当天的租赁时间。

- 地点（where）：维度表dim_storet用来记录DVD光盘是从哪个商店租赁的，和dim_staff、dim_customer一样，dim_store也是缓慢增长维，也有一组列用来记录同一个商店的不同历史版本。

- 事件（what）：这组包括dim_actor和dim_film两个维度表，它们是租赁业务的主题。只有dim_film表和fact_rental表直接关联，因为店员才是租赁和归还的实际对象。但是，一部电影由众多演员构成，这些演员在某种意义上也是租赁对和归还对象。这就是所谓的桥接表dim_film_actor_bridge的由来，该表联系了演员和电影。另外，该表保存了一个权重因子，用来评估一个演员对影片的贡献值。通过原始指标乘以权重因子就可以从演员的角度分析租赁收入，而把原来的指标值看成附加值。
  例如，可以回答这样的问题：上个月Robert De Niro或者Al Pacino的电影获得了多少租金收益？

- 在租赁业务的星型模型中，从源数据库sakila模型中派生出来的每个维度（除了表film_date和dim_time）都对应这sakila数据模型中的每个表。例如，维度表dim_store对应着业务系统中的store表，维度表dim_actor对应这actor表。

键和变更数据捕获：

- 除了表dim_date和dim_time，每个维度表都使用**自增列作为代理主键**，表dim_date和dim_time的主键叫做智能键。这两个表的智能键分别来源于部分时间和日期，可以在ETL中直接发挥作用，也用来对事实表作区分。

- 维度表的键值被用来关联表fact_rental和维度表。所有维度表的主键列都以<维度名称>_key来命名，<维度名称>就是维度表的表名除了dim_前缀之外的剩余部分。

- 源模式中的每个表都有last_uodate字段，该字段保存了一个时间戳TIMESTAMP，用来存储每一行的最后修改（或添加）时间，这个字段对于变更数据捕获非常有用，变更数据捕获对数据持续增长的场景非常有用。变更数据捕获的方法有很多种，这里使用的是一种最直接的方法：每个维度表都有last_update字段，这个字段保存了原始sakila模式中对应表的last_update字段的值。这样可以在维度表上执行一个查询，获得最后加载的日期/时间，并用这个日期/时间来识别和抽取所有源数据库里对应表的最新变更的数据行。

- 除了代理主键，每个维度表也包含一列用来存储来自sakila数据模型的主键值，例如，星型模型中的表dim_film有一列film_id用来保存表film中的film_id，这些列非常重要，是用来判断变更的数据是增加的还是更新的数据。

配置Sakila数据库和业务租赁星型模型数据仓库：

- 运行以下sql文件：