Титульный лист

Реферат

Содержание

Термины и определения

Перечень сокращений и обозначений

ВВЕДЕНИЕ

В компании «ЭР-Телеком» многие статистические данные отслеживаются с помощью OLAP-кубов, состоящих из метрик и измерений. На текущий момент в компании не существует ни одного измерения, построенного на структуре parent-child.

Целью данной работы является доказательство следующей гипотезы:

Можно создать иерархическую структуру parent-child в Oracle Business Intelligence (Oracle BI), которая будет адекватно отображать предметную область "Подразделение" с сохранением ранее созданных связей и данных, а также сможет стать основой при разработке иерархических структур parent-child для других предметных областей компании АО "Эр-Телеком Холдинг".

Далее в тексте «заказчик» равнозначно «компания АО "Эр-Телеком Холдинг"».

Существует достаточно много статей по создании различного типа иерархий в базах данных, но про возможность построения parent-child иерархии упоминается в статье «Oracle BIEE 11g – Parent Child Hierarchies – Multiple Modeling Methods» [1], и в технической документации Oracle было найдено описание построения ключевых объектов и связей для создания parent-child иерархии. [2]. Подробнее об информации в этих источниках говорится в основной части работы.

В отличие от упомянутых источников, в данной работе требуется доказать возможность построения иерархической структуры для предметной области, которая уже существует в Oracle BI, и добавленная структура не должна нарушить имеющиеся связи и должна сохранить имеющиеся данные.

Доказательство гипотезы позволит разрабатывать иерархические структуры parent-child для OLAP-кубов различных предметных областей компании АО "Эр-Телеком Холдинг". Данные разработки позволят снизить трудозатраты аналитиков для построения отчетов и сбора статистики. На текущий момент из имеющихся данных иерархия собирается в Excel, то есть затрачивается время на построение нового отчета из множества других, а затем его анализ. Возможность построение иерархического отчета в Oracle BI позволит избежать ручного построения данных отчетов.

Объект исследования: Измерение «Подразделение», являющееся частью предметной области OLAP-кубов HR.

Предмет исследования: иерархическая структура parent-child.

Для выполнения поставленной целы были выделенные следующие задачи:

1. Проанализировать ПО и инструменты, предоставленные заказчиком для разработки;
2. Проанализировать существующие разработки в этой сфере;
3. Подтвердить или опровергнуть поставленную гипотезу на основе собственных разработок;
4. Описать полученные результаты.

ОСНОВНАЯ ЧАСТЬ

1. **Изучение ПО и инструментов**
   1. СУБД

Данные в компании хранятся в реляционных базах данных, которые обеспечиваются СУБД Oracle.

Реляционная база данных – это связанная информация, представленная в виде двумерных таблиц. Двумерные таблицы состоят из строк и столбцов. У каждой строки есть уникальный идентификатор – это может быть одно поле или несколько – который называется первичным ключом. По нему легко выполнить поиск в таблице одной конкретной строки. Вся информация в базе хранится в виде строк – множества полей, привязанных к конкретному идентификатору, причем число этих полей совпадает с числом столбцов.[3]

Для доступа к реляционным базам данных используется декларативный язык программирования SQL.

SQL (Structured Query Language) — это структурированный язык запросов. На этом языке можно формулировать выражения (запросы), которые извлекают требуемые данные, модифицируют их, создают таблицы и изменяют их структуры, определяют права доступа к данным и многое другое. [4]

СУБД – это программное обеспечение, которое используется для создания и работы с базами данных. Главная функция СУБД – это управление данными (которые могут быть как во внешней, так и в оперативной памяти). СУБД обязательно поддерживает языки баз данных, а также отвечает за копирование и восстановление данных после каких-либо сбоев. [5]

В компании АО «ЭР-Телеком Холдинг» используется СУБД Oracle. В основе большей части Oracle лежит PL/SQL — язык программирования, который предоставляет процедурные расширения используемой в Oracle версии SQL, а также служит языком программирования инструментария Oracle. [6]

СУБД Oracle расположена на двух основных серверах: SA и DWH.

Сервер SA (System Administration) взаимодействует с другими источниками данных, обрабатывает данные, хранит данные короткие промежутки времени. Отсутствует историчность. Лишние данные и таблицы регулярно вычищаются. На этом сервере можно производить основную обработку данных и приведение к необходимому виду.

Сервер DWH позволяет хранить большой объем данных, на нем содержаться исторические данные по многим направлениям деятельности, включая блок продаж, HR-блок, блок выручки/затрат и т.д. Информация на DWH пополняется ежедневно, напрямую записывается в таблицы, без дополнительной обработки. Обработка данных на DWH производится в исключительных случаях.

В одном из источников предложено следующее определение для DWH:

DWH — это система данных, отдельная от оперативной системы обработки данных. В корпоративных хранилищах в удобном для анализа виде хранятся архивные данные из разных, иногда очень разнородных источников. Эти данные предварительно обрабатываются и загружаются в хранилище в ходе процессов извлечения, преобразования и загрузки, называемых ETL. Решения ETL и DWH — это одна система для работы с корпоративной информацией и ее хранения. [7]

* 1. OLAP кубы, метрики, измерения, ETL

Куб OLAP представляет собой структуру данных, которая обеспечивает возможность быстрого анализа данных за рамками ограничений реляционных баз данных. Кубы способны отображать и суммировать большие объемы данных, также предоставляя пользователям доступ к любым точкам данных с возможностью поиска. Таким образом, данные могут быть сведены, фрагментированы и обработаны по мере необходимости для решения самых широкого спектра вопросов, относящихся к интересующей вас области пользователя. [8]

Измерение примерно эквивалентно классу пакета управления. Каждый класс пакета управления имеет набор свойств, а каждое измерение — набор атрибутов, при этом каждый атрибут сопоставляется с одним свойством класса. Измерения позволяют выполнять фильтрацию, группирование и маркировку данных. К примеру, можно отфильтровать компьютеры по установленной операционной системе или сгруппировать людей по категориям, используя пол или возраст. Затем данные могут быть представлены в формате, где данные классифицируются по категориям и категориям, что позволяет более - глубоко анализировать анализ. Измерения также могут иметь естественные иерархии, позволяющие пользователям "детализировать" до более детального уровня детализации. К примеру, измерение даты обладает иерархией, позволяющей выполнять детализацию до уровня лет, затем — до уровней кварталов, месяцев, недель и отдельных дней.

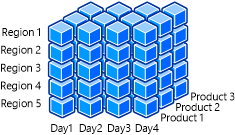


Рисунок 1.2.1 – OLAP куб

В этом рисунке 1.2.1 показан куб OLAP, содержащий измерения даты, региона и продукта.

Меры, метрики — это числовые значения, позволяющие пользователям создавать плоскостные и объемные срезы, выполнять агрегирование и анализ. Они являются одной из основных причин построения кубов OLAP на основании инфраструктуры хранилищ данных. При помощи служб SSAS можно создавать кубы OLAP, использующие бизнес-правила и вычисления для форматирования и отображения мер в настраиваемом формате. Большой объем времени разработки куба OLAP тратится на определение того, какие меры будут отображены, и каким образом они будут вычисляться.

Когда пользователь детализирует данные куба OLAP, он анализирует данные на другом уровне уплотнения. Уровень детальности данных повышается с каждой операцией детализации, что позволяет пользователю изучать данные на разных уровнях иерархии. По мере детализации пользователь переходит от общей информации к данным, имеющим более узкий фокус. [9]

На текущий момент в компании АО «ЭР-Телеком Холдинг» реализована уровневая иерархия. Она принципиально отличается от того, что хотели бы видеть конечные пользователи, а, следовательно, от того, что требуется построить в ходе исследования. В качестве примера ниже на рисунке 1.2.2 представлена текущая уровневая иерархия по измерению «Календарь» OLAP-куба.

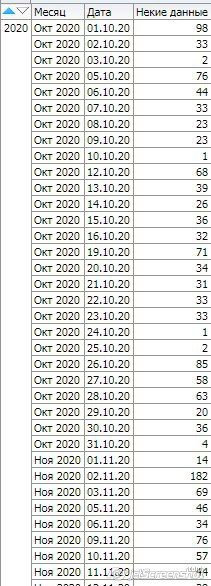


Рисунок 1.2.2 – Данные OLAP-куба в разрезе уровневой иерархии «Календарь»

На рисунке 1.2.2 представлены данные, которые можно посмотреть в разрезе года, месяца, дня. Их можно посмотреть одновременно во всех разрезах, но каждый из них будет выглядеть, как отдельный столбец. Конечные пользователи же хотят видеть иерархию на подобии файловой структуры (рисунок 1.2.3), которую можно свернуть/развернуть и в которой можно посмотреть метрики на каждом уровне иерархии.



Рисунок 1.2.3 – Пример иерархической структуры

На рисунке 1.2.3 представлена иерархическая структура, которая является прототипом того, что конечные пользователи хотят видеть в анализах вместо того, что представлено на рисунке 1.2.2. Видно, что элементы структуры могут содержать элементы нижнего уровня, разворачивать их, и, наоборот, прятать. При этом расчетные метрики должны выводить для каждого уровня корректные значения, включающие как значения самого уровня, так и все внутренних.

* 1. BI

Business Intelligence (BI, Бизнес-аналитика) — это набор IT-технологий для сбора, хранения и анализа данных, позволяющих предоставлять пользователям достоверную аналитику в удобном формате, на основе которой можно принимать эффективные решения для управления бизнес-процессами компании. [10]

Все уровни пользователей, от сотрудников до учредителей, получают гибкий доступ к необходимой им управленческой отчетности, не прибегая к помощи IT-специалистов.

Oracle BI – это основной инструмент для аналитической работы с данными, который используют конечные пользователи, предоставленный компанией АО «ЭР-Телеком Холдинг». Именно с помощью этого инструмента строятся анализы, которые должны будут содержать иерархическую структуру. Заказчиком предоставлено обеспечение взаимодействия BI и серверов баз данных. Для сохранения корректного взаимодействия при описании новой структуры данных необходимо придерживаться регламентов компании.

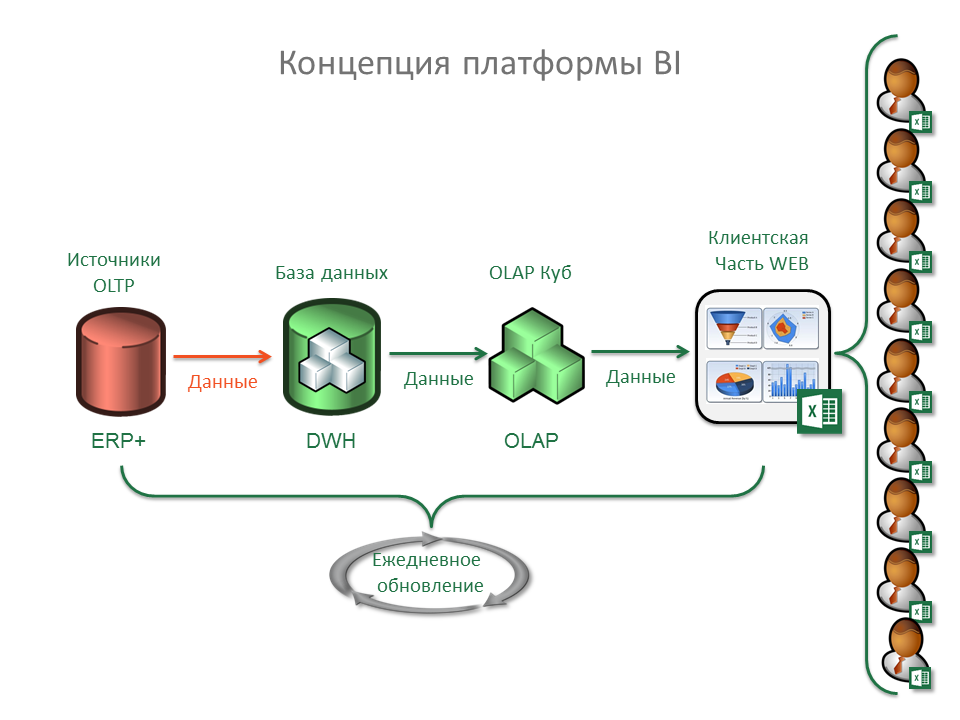
[](http://biweb.ru/wp-content/uploads/2016/09/BI.png)

Рисунок 1.3.1 – Концепция платформы BI

Концепция платформы BI, представленная на рисунке 1.3.1, состоит из следующих компонентов:

* ETL-инструменты: программы, позволяющие выполнять загрузку данных в DWH из различных учетных систем. Заказчиком в качестве ETL-инструмента предоставлена программа Oracle Date Integrator 11g;
* DWH-хранилище: полноценная база данных SQL для подготовки и хранения данных для аналитики;
* OLAP-кубы: технология, позволяющая делать в реальном времени (1-5 секунд) любые отчеты и проводить полноценный анализ данных;
* Клиентские приложения: как правило, для детального анализа данных и построения динамических отчетов пользователи используют сводные таблицы, подключенные к OLAP-кубам. В компании АО «ЭР-Телеком Холдинг» в качестве клиентского приложения выступает Oracle BI.

Основные преимущества BI:

* Скорость построения отчетов;
* Динамический анализ данных в любой детализации;
* Быстрый анализ любого объема данных (технология OLAP);
* Автоматизация подготовки данных для отчетов и построения корпоративной отчетности;
* Консолидация данных (данные для отчетов могут быть в разных учетных системах);
* Анализ показателей План/Факта, анализ выполнения различных KPI;
* Удобная визуализация данных на Web (при этом обновленные данные в отчет поступают автоматически);
* Единый и удобный доступ к аналитической отчетности для всех сотрудников через корпоративный BI портал;
* Уменьшение нагрузки на Учетные системы.

В целом — повышение общей управляемости и эффективности бизнеса. [11]

В компании АО «ЭР-Телеком Холдинг» BI состоит из трех основных уровней, через которые проходят данные:

1. SA – сервер, на котором расположена база данных. Взаимодействует с другими источниками данных, обрабатывает данные, хранит данные кубов только в момент сбора или до следующего сбора. Данные измерений хранятся на сервере полностью для корректного сбора кубов.
2. DWH – сервер, на котором расположена база данных. Предназначен для хранения данных, сюда обращаются анализы BI для получения данных. Умеет быстро анализировать данные, так как на каждой таблице есть индексы и регулярно происходит сбор статистики.
3. BI – клиентская часть, через которую аналитики могут строить анализы. Веб-интерфейс, имеющий в себе информацию о структуре данных на DWH. Позволяет обращаться к DWH за данными напрямую.
   1. Инструменты

Для разработки измерения заказчиком были предоставлены следующие программы: Oracle Data Integrator 11g (ODI), Tool for Oracle Application Developers (TOAD), Oracle Business Intelligence Enterprise Edition (OBIEE). Заказчик строго ограничил использование каких-либо других инструментов, помимо указанных.

Далее приведено краткое описание каждой из используемых программ и их назначение.

Oracle Data Integrator (ODI) — это интеграционная платформа корпоративного уровня, которая обеспечивает извлечение, преобразование и загрузку данных из разнообразных источников: баз данных, файлов и других источников (например, LDAP каталогов или WEB-сервисов). Позволяет создавать автоматические пакеты сбора и обработки данных. [12]

Tool for Oracle Application Developers (TOAD) - инструмент для разработчиков приложений Oracle. Позволяет просматривать базы данных, выполнять PL/SQL код, использовать средства отладки. [13]

Oracle Business Intelligence Enterprise Edition (OBIEE) – инструмент для описания структуры BI. Позволяет описать структуру таблиц, их связи, пути поиска данных при построении анализов. Описанная в АТ структура является основной связи Oracle BI и DWH.

Структура данных в АТ состоит из трех частей:

Физический уровень – здесь хранится полное описание таблиц сервера: название, поля, типы полей, первичные ключи, внешние ключи и так далее.

Бизнес-уровень – здесь хранится описание структур запросов, а именно: к каким таблицам необходимо делать запрос в каком случае, если есть несколько таблиц агрегатов; на какую таблицу связей опираться при создании иерархического запроса; как считать метрики в разрезе измерений; и так далее.

Презентационный уровень – здесь хранится структура в таком виде, в каком видят её пользователи. Сюда выносятся необходимые метрики и измерения, группируются по предметным областям. [14]

1. **Анализ существующих разработок**

На текущий момент в компании АО «ЭР-Телеком Холдинг» нет ни одного примера успешной реализации иерархической структуры в BI. Данная разработка является принципиально новой. На текущий момент в компании АО «ЭР-Телеком Холдинг» реализована уровневая иерархия, представленная на рисунке 1.2.2.

Для решения поставленной задачи было исследовано множество литературы. В статье «Oracle BIEE 11g – Parent Child Hierarchies – Multiple Modeling Methods» [1] продемонстрировано возможное решение поставленной задачи через иерархическую структуру в OBIEE. Описанная в статье структура не удовлетворяет условиям, поставленным регламентами компании, но доказывает, что существует возможность решения поставленной задачи.

Однако в статье не указано на чем строится данная иерархическая структура. В технической документации Oracle было найдено описание построения ключевых объектов и связей для создания parent-child иерархии. [2]

Таким образом, уже доказано, что parent-child иерархию возможно построить. Это подразумевает создание многоуровневой иерархии, где каждый элемент, кроме корневых, имеет вышестоящий элемент. В отличие от уровневой иерархий, все элементы измерения иерархии parent-child находятся в одном логическом столбце.

В иерархии parent-child родительский элемент элемента находится в другой строке того же логического столбца, на который указывает родительский ключ. Это отличается от иерархии на основе уровней, где родительский элемент элемента находится в другом логическом столбце в той же строке. Другими словами, навигация в иерархии parent-child следует за значениями данных, а навигация в иерархии на основе уровней следует за структурой метаданных.

В рамках этой работы необходимо доказать, что иерархию parent-child можно построить для предметной области «Подразделение», которая уже существует в Oracle BI и имеет ряд таблиц в базе данных, и добавленная структура не должна нарушить имеющиеся связи и должна сохранить имеющиеся данные.

1. **Разработка иерархической структуры для доказательства поставленной гипотезы**

В рамках доказательства гипотезы была проведена разработка иерархического измерения parent-child для измерения «Подразделение».

* 1. Входные данные

Рассмотрим имеющиеся данные. Основой разработки является существующее измерение «Подразделение». На текущий момент данные для сбора этого измерения приходят в .csv-файле из сторонней системы (1С). Данный файл содержит следующие поля:

* Date – дата выгрузки данных;
* TypicaSubdivision – название типового подразделения;
* TypicaSubdivision\_guid - уникальный идентификатор типового подразделения;
* Organization – название организации, к которой относится типовое подразделение;
* Subdivision – название подразделения, которое относится к этому типовому подразделению;
* Subdivision\_guid - уникальный идентификатор подразделения;
* Basic – является ли типовое подразделение базовым;
* Positions – должность, которая относится к этому типовому подразделению;
* Positions\_guid - уникальный идентификатор должности.

Ниже представлен пример того, как выглядят данные в исходном файле.

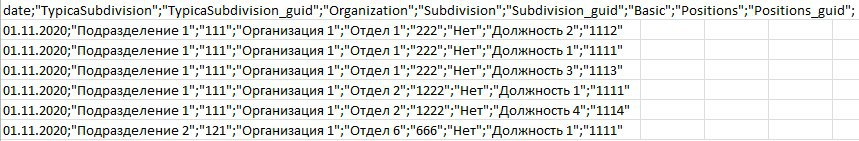


Рисунок 3.1.1 – Исходные данные

Как видно из рисунка 3.1.1, о каждом подразделении приходит информация, куда оно входит: организация; и что в него входит: отдел и должность. Эти данные необходимы для привязки измерения к кубам, у которых нет информации о подразделении, но есть информация об отделах, организациях и должностях. Сочетание этих трех показателей имеет однозначную привязку к подразделению.

Для реализации задачи необходимо добавить в выгрузку следующие поля:

* TypicalSubdivisionParent - название родительского типового подразделения;
* TypicalSubdivisionParent\_guid – уникальный идентификатор.

Однако в рамках реализации задачи был рассмотрен следующий вопрос: если в изначальной выгрузке каждое типовое подразделение дублируется множество раз, то, возможно, будет удобнее создать новую выгрузку, содержащую только информацию об иерархии, чем добавлять поля в старую выгрузку? Приблизительный подсчет показал следующее: строк в изначальной выгрузке порядка 300 тысяч, в то время как Подразделений около 1 тысячи. Соответственно создание новой выгрузки, состоящей только из иерархии (даже с учетом дополнительных данных, например, даты), будет занимать примерно в 120 раз меньше ресурсов, чем дополнение существующей выгрузки. Также был рассмотрен вариант дополнять только одну строку в изначальной выгрузке по каждому подразделению, но в этом случае тратятся ресурсы на стороне BI для обработки и поиска каждой такой строки.

В конечном итоге сделан следующий вывод: для создания корректно работающего иерархического измерения с минимальными затратами памяти необходимо сформировать новую выгрузку со следующими полями:

* Date – дата выгрузки данных;
* TypicaSubdivision – название подразделения;
* TypicaSubdivision\_guid - уникальный идентификатор подразделения;
* TypicaSubdivisionParent - название родительского подразделения;
* TypicaSubdivisionParent\_guid – уникальный идентификатор.
  1. Структура данных

Рассмотрим, какие необходимо предпринять шаги для создания подходящих таблиц в базах данных на серверах SA и DWH.

Известно, что сбор измерения состоит из получения данных, добавления данных в таблицы внутри базы данных, перенос на DWH.

В ходе исследования иерархической структуры было выяснено, что для создания новой структуры необходимо:

* Изменить таблицы, куда поступают и где обрабатываются данные таким образом, чтобы получать новые данные об иерархии;
* Изменить таблицы хранения данных таким образом, чтобы в измерении было дополнительное поле, обозначающее родителя;
* Изменить описание структуры в BI таким образом, чтобы BI имел возможность создавать иерархические запросы. [15]

Для того, чтобы BI смог простраивать иерархический запрос, необходима еще одна таблица: таблица связей между элементами, в которой для каждого элемента указывается родитель-дистанция-признак\_листа для каждого вышестоящего элемента в дереве. Поля MEMBER\_KEY, ANCESTOR\_KEY, DISTANCE, IS\_LEAF. На основе этой таблицы BI сможет от каждого элемента прийти к любому вышестоящему родителю.

Сбор таблицы связей является самым сложным шагом, так как требует больших ресурсов. Для сбора необходимо будет написать и оптимизировать отдельную процедуру, способную для каждого элемента выявить вышестоящих родителей.

Сбор измерения должен происходить в автоматическом режиме. Это реализуется с помощью инструмента ODI. Процедура заполнения таблицы связей должны быть включена в ежедневный сбор.

Описание структуры данных для BI реализуется с помощью инструмента AdminTools. Необходимо указать, что измерение является иерархическим, по каким полям происходит привязка к иерархии, и вынести измерение на презентационный уровень.

**Вывод:** Ключевой момент, который был выяснен в течение изучения предметной области: иерархию parent-child BI поддерживает и её можно настроить на бизнес уровне AdminTools, но для этого нужна особая таблица связей, и перестройка основной таблицы.

В рамках этого необходимо создать новую выгрузку данных, где будет приходить связь parent-child. Необходимо переработать существующие таблицы сбора измерения и создать новую таблицу связей. Необходимо вынести иерархия на презентационный уровень BI.

* 1. Разработка иерархического измерения

#### Автоматизация сбора

Из внешней системы был сгенерирован файл с новой выгрузкой по требуемым условиям. Также была создана внешняя таблица, в которую попадают приходящие данные.

Была проведена разработка автоматического сбора с помощью ODI для ежедневной загрузки данных. В результате был получен пакет сбора, представленный на рисунке 3.3.1.

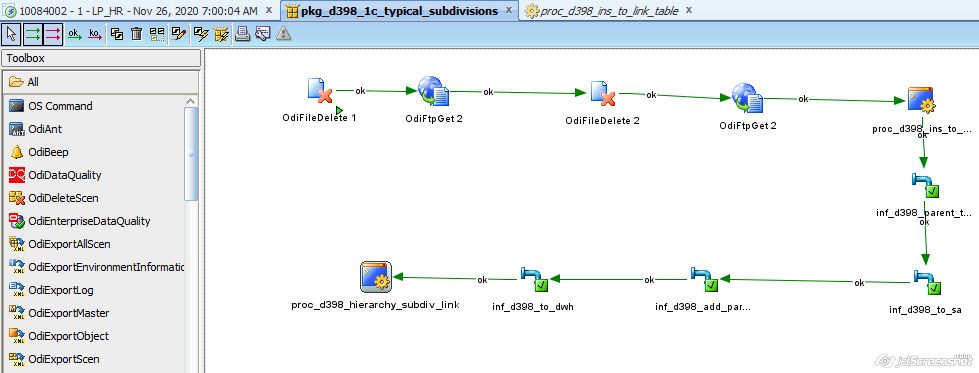


Рисунок 3.3.1 – Новый пакет сбора измерения

Пакет сбора измерения, приведенный на рисунке 3.3.1, состоит из десяти шагов:

* 1. Удаление первого файла с данными на сервере SA за прошлый день;
  2. Перенос нового первого файла с FTP на сервер SA;
  3. Удаление второго файла с данными на сервере SA за прошлый день;
  4. Перенос нового второго файла с FTP на сервер SA;
  5. Заполнение таблицы трансляций (требуется для дальнейшей привязки кубов к этому измерению с помощью должностей и организаций). Процедура сбора из внешней таблицы.
  6. Перенос данных в промежуточную таблицу (регламент запрещает сбор измерения напрямую из внешней таблицы);
  7. Сбор измерения на SA из таблицы трансляций. Пополняется измерение новыми данными.
  8. Добавление в собранное измерение данных о родителях подразделений;
  9. Перенос новых данных в таблицу куба на DWH;
  10. Процедура сбора таблицы связей на DWH с предварительным очищением этой таблицы.

Главным элементом в описанном пакете для доказательства гипотезы является шаг 10 – сбор таблицы связей на сервере DWH. На данном этапе собирается таблица связей между элементами, в которой для каждого элемента указывается родитель-дистанция-признак\_листа для каждого вышестоящего элемента в дереве. Это поля MEMBER\_KEY, ANCESTOR\_KEY, DISTANCE, IS\_LEAF. На основе этой таблицы BI сможет от каждого элемента прийти к любому вышестоящему родителю. Сбор реализован с помощью написанной процедуры. Ниже представлено описание процедуры.

Для начала достаточно узнать количество уровней, которое будет в иерархии, и записать их в переменную. Так будет известно, сколько раз необходимо спуститься на следующий уровень рекурсии.

Листинг 3.5 – Поиск количества уровней иерархии

declare

v\_max\_depth integer;

begin

select max(level) into v\_max\_depth

from D\_1C\_TYPICAL\_SUBDIVISIONS

connect by prior DIMENSION\_KEY=TYPICAL\_SUBDIVISION\_PARENT

start with TYPICAL\_SUBDIVISION\_PARENT is null;

end;

Дальше можно будет кусочно собрать запрос с помощью цикла.

Листинг 3.6 – Пример цикличной конструкции запроса

for i in 1..v\_max\_depth - 1 loop

...

end loop;

Но кусочный проход по таблице не слишком удобен, так как после каждого нужно использовать commit, а это может привести к пропуску ошибок где-то в середине заполнения таблицы. Поэтому было принято решение использовать динамический PL/SQL и рекурсивно собрать один большой запрос, который дальше можно выполнить за один раз.

Ниже представлен весь запрос полностью.

Листинг 3.7 – Полный код сбора таблицы связей

declare

v\_max\_depth integer;

v\_stmt varchar2(32000);

i integer;

begin

select max(level) into v\_max\_depth

from D\_1C\_TYPICAL\_SUBDIVISIONS

connect by prior DIMENSION\_KEY=TYPICAL\_SUBDIVISION\_PARENT

start with TYPICAL\_SUBDIVISION\_PARENT is null;

*-- Этот кусок посчитал, сколько всего уровней в иерархии.*

v\_stmt := 'insert into D398\_HIERARCHY\_D\_SUBD\_LINK (MEMBER\_KEY, ANCESTOR\_KEY, DISTANCE, IS\_LEAF)' || chr(10)

|| 'select DIMENSION\_KEY as member\_key, null, null, 0 from D\_1C\_TYPICAL\_SUBDIVISIONS where TYPICAL\_SUBDIVISION\_PARENT is null' || chr(10)

|| 'union all' || chr(10)

|| 'select' || chr(10)

|| ' member\_key,' || chr(10)

|| ' replace(replace(ancestor\_key, ''\p'', ''|''), ''\'', ''\'') as ancestor\_key,' || chr(10)

|| ' case when depth is null then 0' || chr(10)

|| ' else max(depth) over (partition by member\_key) - depth + 1' || chr(10)

|| ' end as distance,' || chr(10)

|| ' is\_leaf' || chr(10)

|| 'from' || chr(10)

|| '(' || chr(10)

|| ' select' || chr(10)

|| ' member\_key,' || chr(10)

|| ' depth,' || chr(10)

|| ' case' || chr(10)

|| ' when depth is null then '''' || member\_key' || chr(10)

|| ' when instr(hier\_path, ''|'', 1, depth + 1) = 0 then null' || chr(10)

|| ' else substr(hier\_path, instr(hier\_path, ''|'', 1, depth) + 1, instr(hier\_path, ''|'', 1, depth + 1) - instr(hier\_path, ''|'', 1, depth) - 1)' || chr(10)

|| ' end ancestor\_key,' || chr(10)

|| ' is\_leaf' || chr(10)

|| ' from' || chr(10)

|| ' (' || chr(10)

|| ' select DIMENSION\_KEY as member\_key, TYPICAL\_SUBDIVISION\_PARENT as ancestor\_key, sys\_connect\_by\_path(replace(replace(DIMENSION\_KEY, ''\'', ''\''), ''|'', ''\p''), ''|'') as hier\_path,' || chr(10)

|| ' case when DIMENSION\_KEY in (select TYPICAL\_SUBDIVISION\_PARENT from D\_1C\_TYPICAL\_SUBDIVISIONS ) then 0 else 1 end as IS\_LEAF' || chr(10)

|| ' from D\_1C\_TYPICAL\_SUBDIVISIONS ' || chr(10)

|| ' connect by prior DIMENSION\_KEY = TYPICAL\_SUBDIVISION\_PARENT ' || chr(10)

|| ' start with TYPICAL\_SUBDIVISION\_PARENT is null' || chr(10)

|| ' ),' || chr(10)

|| ' (' || chr(10)

|| ' select null as depth from dual' || chr(10);

for i in 1..v\_max\_depth - 1 loop

v\_stmt := v\_stmt || ' union all select ' || i || ' from dual' || chr(10);

end loop;

v\_stmt := v\_stmt || ' )' || chr(10)

|| ')' || chr(10)

|| 'where ancestor\_key is not null' || chr(10);

execute immediate v\_stmt;

end;

Задается изначальное значение переменной v\_stmt, в которой будет храниться весь запрос.

Листинг 3.8 – Поиск путей от каждой вершины до корневой

select DIMENSION\_KEY as member\_key, TYPICAL\_SUBDIVISION\_PARENT as ancestor\_key,

sys\_connect\_by\_path(replace(replace(DIMENSION\_KEY, '\', '\'), '|', '\p'), '|') as hier\_path,

case when DIMENSION\_KEY in (select TYPICAL\_SUBDIVISION\_PARENT from dev\_bitest.D\_1C\_TYPICAL\_SUBDIVISIONS ) then 0 else 1 end as IS\_LEAF

from dev\_bitest.D\_1C\_TYPICAL\_SUBDIVISIONS

connect by prior DIMENSION\_KEY = TYPICAL\_SUBDIVISION\_PARENT

start with TYPICAL\_SUBDIVISION\_PARENT is null

Представленный выше фрагмент достает путь от каждой вершины до корневой вершины. Является ли вершина листом определяется так: если нет ни одной вершины, для которой она была бы родителем, то это листовой элемент. Результат запроса выглядит следующим образом.

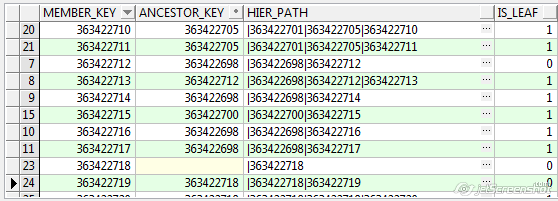


Рисунок 3.3.2 – Результат запроса

На рисунке 3.3.2 представлен результат описанного выше запроса. Для каждой вершины определен путь от корневой вершины до неё.

Ниже представлен фрагмент кода, который показывает, каким образом есть возможность опуститься до выбранного элемента, сохраняя все пройденные узлы.

Листинг 3.9 – Рекурсивный спуск по уровням глубины

sys\_connect\_by\_path – это рекурсивный запрос, которому задаются условия:

connect by prior DIMENSION\_KEY = TYPICAL\_SUBDIVISION\_PARENT

start with TYPICAL\_SUBDIVISION\_PARENT is null

Отдельно с помощью цикла получаем все возможные уровни глубины. Ниже представлен фрагмент, расписанный так, как он выглядит в момент запроса. Выше в полном запросе указано, как выглядит данный фрагмент в коде.

Листинг 3.10 – Фрагмент, который заполняется в цикле

select null as depth from dual

union all select 1 from dual

union all select 2 from dual

union all select 3 from dual

union all select 4 from dual…

Результат запроса представлен на рисунке ниже.

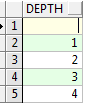


Рисунок 3.3.3 – Результат запроса

На рисунке 3.3.3 представлен результат запроса уровней глубины. При полном выполнении кода уровней глубины будет равно количеству уровней иерархии. Далее результаты двух запросов собираются в один.

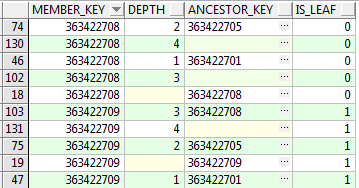


Рисунок 3.3.4 – Результат запроса

На рисунке 3.3.4 представлено совмещение двух результатов запроса в один. Для каждого элемента в запросе получается число строк по числу строк в таблице глубины, где на каждом уровне глубины написано, с каким родительским элементом этом уровне связан элемент. Элемент связан с самим собой там, где глубина null. Там, где глубина равна 1, указан вышестоящий элемент для этого элемента. Чем больше значение глубины, тем дальше находится вышестоящий элемент.

В качестве примера рассмотрим элемент 363422709, для которого цепочка связей в иерархии равна |363422701|363422705|363422708|363422709, где 363422701– корневой элемент.

Как видно из рисунка 3.3.4, связь элементов 363422701 и 363422709 стоит с глубиной 1, так как в данном случае под глубиной подразумевается количество уровней от корневого элемента. Но в таблице связей глубиной должно быть другое значение: насколько элементов далеко данный элемент от выбранного родителя (в данном примере на 3). Необходимые данные можно получить с помощью следующего запроса:

Листинг 3.11 – Получение глубины для каждой вершины от каждой предыдущей

select

member\_key,

replace(replace(ancestor\_key, '\p', '|'), '\', '\') as ancestor\_key,

case when depth is null then 0

else max(depth) over (partition by member\_key) - depth + 1

end as distance,

is\_leaf

from …

Где источником данных являются результаты предыдущих запросов, а именно, результат запроса с рисунка 3.3.4.

В конечном итоге получаются следующие данные, которые можно вставлять в таблицу связей.

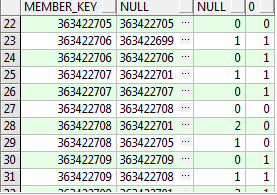


Рисунок 3.3.5 – Результат запроса

На рисунке 3.3.5 представлен пример данных, которые заносятся в таблицу связей.

**Вывод**: на серверах баз данных, куда обращается BI для получений данных, можно реализовать хранение данных в том виде, в котором необходимо для дальнейшего построения в BI иерархических запросов по структуре parent-child.

* 1. Внесение данных в структуру BI

Следующим шагом необходимо оформить структуру данных в OBIEE. Описание структуры данных в BI представлено в разделе 1.4.

**Физический слой**

Изначальная таблица измерения на физическом слое выглядела следующим образом.

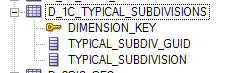


Рисунок 3.4.1 – Старая структура таблицы

Как видно из рисунка 3.4.1, старая структура таблицы измерения состояла из трех полей. Так как в новую структуру добавлено еще одно поле, необходимо его внести и в описание.

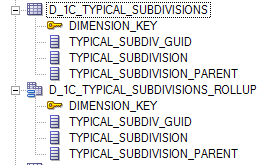


Рисунок 3.4.2 – Новая структура таблицы

Помимо добавления нового поля, был создан алиас измерения. Это сделано для того, чтобы можно было получать значения, как раньше, и чтобы можно было получать значения иерархично. Это важно для кубов, в которых необходимо считать данные в разрезе одного подразделения. При подсчете данных через иерархичное измерение в вышестоящее попадут все данные из нижестоящих в том числе.

Так же была добавлена таблица связей, у которой в качестве внешнего ключа указана связь ancestor\_key к d\_1c\_typical\_subdivision.dimension\_key.

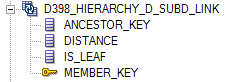


Рисунок 3.4.3 – Структура таблицы связей

На рисунке 3.4.3 представлена структура созданной таблицы связей, которая будет использоваться для построения анализов.

**Бизнес-уровень**

Изначальная структура измерения на бизнес-слое выглядела следующим образом.

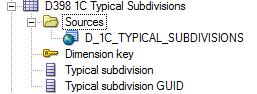


Рисунок 3.4.4 – Старая структура таблицы

Как видно из рисунка 3.4.4, в структуре указываются таблицы-источники и поля, которые необходимо вынести дальше на презентационный уровень. После изменений таблиц источников стало две, поэтому в структуре также указываются обе.

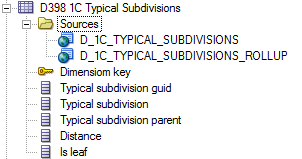


Рисунок 3.4.5 – Измененная структура таблицы

На рисунке 3.25 представлена измененная структура таблицы с указанием новых источников данных. При этом внутри источника с иерархией необходимо указать связь с таблицей связей. Достаточно сделать JOIN с таблицей связей, так как на физическом уровне уже указано, как именно они связываются.

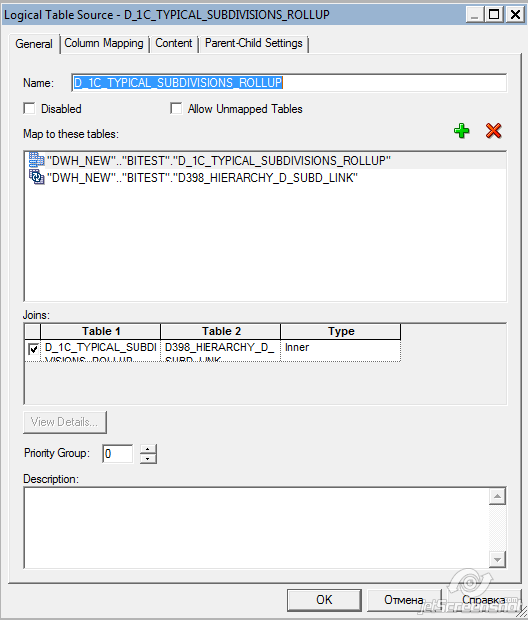


Рисунок 3.4.6 – JOIN с таблицей связей

На рисунке 3.4.6 представлено, как именно связываются между собой иерархическая таблица данных и таблица связей.

На этом же уровне указывается наличие иерархии.

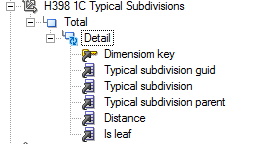


Рисунок 3.4.7 – Структура иерархии

Как видно из рисунка 3.4.7, на бизнес-уровне указывается структура иерархии, где определены уровни, ключи, поля и внутренние связи.

**Презентационный уровень**

Данные переносятся на презентационный уровень, откуда их можно будет доставать для создания анализов. Это выглядит следующим образом.

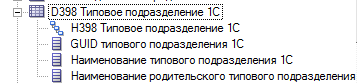


Рисунок 3.4.8 – Структура измерения на презентационном уровне

На рисунке 3.4.8 представлена структура измерения на презентационном уровне. Именно в таком виде структуру увидит аналитик при построении анализов.

**Результат в BI**

После создания полной структуры измерения на трех уровнях OBIEE можно посмотреть, как данные будут представлены в BI. При вынесении данных в анализ полученная иерархия будет выглядеть следующим образом.

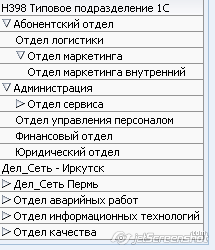


Рисунок 3.4.9 – Результат структуры в анализе

На рисунке 3.4.9 представлены обработанные данные, собранные в иерархическую структуру. Нажимая на стрелочки можно развернуть внутренние части иерархии. При разбиении метрик по данной структуре можно будет увидеть количественные значения данных в разрезе иерархии.

**Вывод**: Было разработано, создано и автоматизировано измерение «Подразделение». Результатом работы стал пакет сбора измерения и структура данных в BI, которую можно использовать для создания анализов. Полученная структура данных соответствует той, которая указана в гипотезе, при этом целостность предметной области не нарушена, изначальные данные сохранены. Гипотезу можно считать доказанной.

ЗАКЛЮЧЕНИЕ

В результате приведенных исследований было разработано, создано и автоматизировано измерение «Подразделение». Результатом работы стал пакет сбора измерения и иерархическая структура данных в BI, которую можно использовать для создания анализов. Полученная структура данных соответствует той, которая указана в гипотезе, при этом целостность предметной области не нарушена, изначальные данные сохранены.

Проведенные исследования показали, что подобную разработку можно повторить для других измерений различных предметных областей компании АО «Эр-Телеком». Гипотезу можно считать доказанной.

Поставленные задачи выполнены, цель достигнута. Разработка внедрена в компанию с целью автоматизации построения иерархических анализов и сокращения трудозатрат.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. RittManMead / Oracle BI EE 11g – Parent Child Hierarchies – Multiple Modeling Methods [Электронный ресурс] URL: <https://www.rittmanmead.com/blog/2010/11/oracle-bi-ee-11g-parent-child-hierarchies-multiple-modeling-methods/> (Дата обращения: 01.03.2021)
2. Oracle / Fusion Middleware Metadata Repository Builder's Guide for Oracle Business Intelligence Enterprise Edition / Fusion Middleware Metadata Repository Builder's Guide for Oracle Business Intelligence Enterprise Edition [Электронный ресурс] URL: https://docs.oracle.com/middleware/12211/biee/BIEMG/GUID-0424E7A7-C7DB-447A-B0C4-0BD6790888EA.htm#hpp\_l\_value\_dimension (Дата обращения: 01.03.2021)
3. Грабер, Мартин SQL для простых смертных / Мартин Грабер. - М.: ЛОРИ, 2014. - 378 c.
4. Дунаев, В. В. Базы данных. Язык SQL для студента / В.В. Дунаев. - М.: БХВ-Петербург, 2017. - 288 c.
5. Community / Базы данных и СУБД [Электронный ресурс] URL: https://timeweb.com/ru/community/articles/bazy-dannyh-i-subd-1 (Дата обращения: 25.11.2020)
6. Прибыл, Билл Oracle PL/SQL. Для профессионалов / Билл Прибыл. - М.: Питер, 2014. - 1020 c.
7. Завтра облачно / Что такое DWH и почему без них данные компании почти бесполезны [Электронный ресурс] URL: https://mcs.mail.ru/blog/chto-takoe-dwh-i-pochemu-bez-nih-dannye-kompanii-bespolezny (Дата обращения: 12.12.2020)
8. Карлова Е.А., Багаева А.П. OLAP-технологии // Актуальные проблемы авиации и космонавтики. 2013. №9. URL: https://cyberleninka.ru/article/n/olap-tehnologii (дата обращения: 12.03.2021).
9. Общие сведения о кубах OLAP в Service Manager для расширенной аналитики [Электронный ресурс] URL: <https://docs.microsoft.com/ru-ru/system-center/scsm/olap-cubes-overview?view=sc-sm-1711> (Дата обращения: 25.11.2020)
10. BI Блог / Что такое BI (Business Intelligence) и зачем он нужен? [Электронный ресурс] URL: <http://biweb.ru/chto-takoe-business-intelligence.html> (Дата обращения: 25.11.2020)
11. Navid Mahlouji. Information Technologies for Business Intelligence// CiteseerX. – 2014. – С. 102 / [Электронный ресурс]. – Режим доступа: URL: https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.978.7425 (Дата обращения: 02.03.2021)
12. ФОРС/Oracle Data Integrator [Электронный ресурс] URL: <https://www.fors.ru/business-solutions/analytical-systems-and-data-warehouse/oracle-data-integrator/> (Дата обращения: 25.11.2020)
13. DEEPEDIT!/ TOAD [Электронный ресурс] URL: <http://deepedit.ru/toad.html> (Дата обращения: 25.11.2020)
14. OLAP.ru / Oracle Business Intelligence – обзор [Электронный ресурс] URL: http://www.olap.ru/home.asp?artId=2344 (Дата обращения: 15.02.2021)
15. Creating a Repository Using the Oracle Business Intelligence Administration Tool [Электронный ресурс] URL: <https://www.oracle.com/webfolder/technetwork/tutorials/obe/fmw/bi/bi1113/biadmin11g_01/biadmin11g.htm#t9> (Дата обращения: 02.03.2021)