ДОДАТОК А

Програмна реалізація системи

```
CREATE TABLE [dbo].[Services_raw](
	[Service Number] [varchar](100) NULL,
	[AdmissionDxCode] [varchar](100) NULL,
	[AdmissionDxCodeDesc] [varchar](100) NULL,
	[AdmitDate] [varchar](100) NULL,
	[AdmitSourceCode] [varchar](100) NULL,
	[AdmitSourceCodeDesc] [varchar](100) NULL,
	[AdmitTime] [varchar](100) NULL,
	[AdmitTypeCode] [varchar](100) NULL,
	[AdmitTypeCodeDesc] [varchar](100) NULL,
	[CurrentBalance] [varchar](100) NULL,
	[DischargeDate] [varchar](100) NULL,
	[DischargeDisposition] [varchar](100) NULL,
	[DischargeDispositionDesc] [varchar](100) NULL,
	[DischargeMonthandYear] [varchar](100) NULL,
	[DischargeTime] [varchar](100) NULL,
	[EntityCode] [varchar](100) NULL,
	[EntityDesc] [varchar](100) NULL,
	[ERLevelCode] [varchar](100) NULL,
	[FinalBillDate] [varchar](100) NULL,
	[DischargeFiscalQuarter] [varchar](100) NULL,
	[DischargeFiscalYear] [varchar](100) NULL,
	[HospitalServiceCode] [varchar](100) NULL,
	[InitialBillDate] [varchar](100) NULL,
	[LengthofService(LOS)] [varchar](100) NULL,
	[ClientAge] [varchar](100) NULL,
	[ClientClassification] [varchar](100) NULL,
	[ClientFirstName] [varchar](100) NULL,
	[ClientLastName] [varchar](100) NULL,
	[ClientStatus] [varchar](100) NULL,
	[ClientType] [varchar](100) NULL,
	[ClientTypeDesc] [varchar](100) NULL,
	[ClientZip] [varchar](100) NULL,
	[TYPE] [varchar](100) NULL,
	[ServiceLine] [varchar](100) NULL,
	[TotalAdjustments] [varchar](100) NULL,
	[TotalCharges] [varchar](100) NULL,
	[TotalPayments] [varchar](100) NULL,
	[ProductLine] [varchar](100) NULL,
	[ClientBirthDate] [varchar](100) NULL,
	[ClientLocation] [varchar](100) NULL,
	[ClientRace] [varchar](100) NULL,
	[ClientRegistrationLocation] [varchar](100) NULL,
	[Service] [varchar](100) NULL,
	[ServiceGroup] [varchar](100) NULL,
	[ImportFileName] [varchar](100),
	[LoadDate][datetime] default GetDate()
)
GO


CREATE TABLE [dbo].[AdjtTran_raw](
	[ServiceNumber] [varchar](100) NULL,
	[EntityCode] [varchar](100) NULL,
	[EntityDesc] [varchar](100) NULL,
	[FacilityCode] [varchar](100) NULL,
	[FacilityDesc] [varchar](100) NULL,
```

```
	[PaymentDate] [varchar](100) NULL,
	[TxnAmount] [varchar](100) NULL,
	[TxnCategory] [varchar](100) NULL,
	[TxnCode] [varchar](100) NULL,
	[TxnCodeDesc] [varchar](100) NULL,
	[TxnPostDate] [varchar](100) NULL
)
GO

CREATE TABLE [dbo].[Charges_raw](
	[ServiceNumber] [varchar](100) NULL,
	[TotalChargeAmount] [varchar](100) NULL,
	[ChargeCode] [varchar](100) NULL,
	[ChargeCodeDesc] [varchar](100) NULL,
	[ChargeDepartment] [varchar](100) NULL,
	[ChargeDeptDesc] [varchar](100) NULL,
	[ChargePostDate] [varchar](100) NULL,
	[ChargeQuantity] [varchar](100) NULL,
	[ChargeServiceDate] [varchar](100) NULL,
	[Quantity] [varchar](100) NULL,
	[FacilityCode] [varchar](100) NULL,
	[FacilityDesc] [varchar](100) NULL,
	[EntityCode] [varchar](100) NULL,
	[EntityDesc] [varchar](100) NULL,
	[RevenueCode] [varchar](100) NULL,
	[RevenueCodeDesc] [varchar](100) NULL,
	[UnitPrice] [varchar](100) NULL
)

GO

CREATE TABLE [dbo].[Services_stg](
	[ID_col] [int] IDENTITY(1,1) NOT NULL,
	[ServiceNumber] [varchar](12) NULL,
	[ServiceAgeByDischarge][int]NULL,
	[AgingID][int]NULL,
	[PostDate][date]NULL,
	[PostDateID][int]NULL,
	[AdmitDate] [date] NULL,
	[AdmitDateID] [int] NULL,
	[AdmitSourceCode] [varchar](6) NULL,
	[AdmitSourceCodeDesc] [varchar](30) NULL,
	[AdmitTime] [datetime] NULL,
	[AdmitTypeCode] [varchar](2) NULL,
	[AdmitTypeDescription] [varchar](9) NULL,
	[CurrentBalance] [money] NULL,
	[ThresholdID] [tinyint] NULL,
	[DischargeDate] [date] NULL,
	[DischargeDateID] [int] NULL,
	[DischargeDisposition] [varchar](10) NULL,
	[DischargeDispositionDesc] [varchar](30) NULL,
	[DischargeTime] [time](7) NULL,
	[EntityCode] [varchar](4) NULL,
	[EntityDesc] [varchar](34) NULL,
	[ERLevelCode] [int] NULL,
	[FacilityCode] [varchar](4) NULL,
	[FacilityDesc] [varchar](30) NULL,
	[FinalBillDate] [date] NULL,
	[FinalBillDateID] [int] NULL,
	[InitialBillDate] [date] NULL,
	[LineofBusiness] [varchar](15) NULL,
	[ClientAge] [varchar](7) NULL,
	[ClientFirstName] [varchar](25) NULL,
	[ClientLastName] [varchar](25) NULL,
```

```
	[ClientStatus] [char](2) NULL,
	[ClientTypeCode] [varchar](7) NULL,
	[ClientTypeID] [int] NULL,
	[ClientTypeDesc] [varchar](20) NULL,
	[ClientCodeZip] [varchar](10) NULL,
	[TYPE][VARCHAR](5),
	[ServiceLine] [varchar](9) NULL,
	[TotalAdjustments] [money] NULL,
	[TotalCharges] [money] NULL,
	[TotalPayments] [money] NULL,
	[Variance] [varchar](25) NULL,
	[LocationCode] [varchar](10) NULL,
	[PatientRace] [varchar](30) NULL,
	[RegistrationLocation] [varchar](10) NULL,
	[Service] [varchar](40) NULL,
	[ServiceGroup] [varchar](22) NULL,
)
GO

CREATE TABLE [dbo].[AdjtTran_stg](
	[Id_col] [int] IDENTITY(1,1) NOT NULL,
	[ServiceNumber] [varchar](12) NULL,
	[EntityCode] [varchar](4) NULL,
	[EntityDesc] [varchar](40) NULL,
	[FacilityCode] [varchar](30) NULL,
	[Facility] [varchar](30) NULL,
	[TxnServiceDate] [date] NULL,
	[TxnServiceDateID] [int] NULL,
	[Amount] [money] NULL,
	[TransactionType] [varchar](10) NULL,
	[TxnCode] [varchar](15) NULL,
	[TxnCodeID] [int] NULL,
	[TxnCodeDesc] [varchar](80) NULL,
	[PostDate] [date] NULL,
	[PostDateID] [int] NULL
)
GO

CREATE TABLE [dbo].[Charges_stg](
	[ID_col] [int] IDENTITY(1,1) NOT NULL,
	[ServiceNumber] [varchar](12) NULL,
	[Amount] [money] NULL,
	[ChargeCode] [varchar](10) NULL,
	[ChargeCodeID] [int] NULL,
	[ChargeDepartment] [varchar](7) NULL,
	[ChargeDeptDesc] [varchar](30) NULL,
	[PostDate] [date] NULL,
	[PostDateID] [int] NULL,
	[Quantity] [smallint] NULL,
	[TxnServiceDate] [date] NULL,
	[TxnServiceDateID] [int] NULL,
	[FacilityCode] [varchar](4) NULL,
	[Facility] [varchar](30) NULL,
	[EntityCode] [varchar](4) NULL,
	[EntityDesc] [varchar](40) NULL,
	[UnitPrice] [money] NULL
)
GO

CREATE TABLE [dbo].[Service_fact](
	[ID_col] [int] IDENTITY(1,1) NOT NULL,
	[ServiceNumber] [varchar](12) NULL,
	[ServiceAgeByDischarge][int]NULL,
	[AgingID][int]NULL,
```

```
	[PostDate][date]NULL,
	[PostDateID][int]NULL,
	[AdmitDate] [date] NULL,
	[AdmitDateID] [int] NULL,
	[AdmitSourceCode] [varchar](6) NULL,
	[AdmitSourceCodeDesc] [varchar](30) NULL,
	[AdmitTime] [datetime] NULL,
	[AdmitTypeCode] [varchar](2) NULL,
	[AdmitTypeDescription] [varchar](9) NULL,
	[CurrentBalance] [money] NULL,
	[ThresholdID] [tinyint] NULL,
	[DischargeDate] [date] NULL,
	[DischargeDateID] [int] NULL,
	[DischargeDisposition] [varchar](10) NULL,
	[DischargeDispositionDesc] [varchar](30) NULL,
	[DischargeTime] [time](7) NULL,
	[EntityCode] [varchar](4) NULL,
	[EntityDesc] [varchar](34) NULL,
	[ERLevelCode] [int] NULL,
	[FacilityCode] [varchar](4) NULL,
	[FacilityDesc] [varchar](30) NULL,
	[FinalBillDate] [date] NULL,
	[FinalBillDateID] [int] NULL,
	[InitialBillDate] [date] NULL,
	[LineofBusiness] [varchar](15) NULL,
	[ClientAge] [varchar](7) NULL,
	[ClientFirstName] [varchar](25) NULL,
	[ClientLastName] [varchar](25) NULL,
	[ClientStatus] [char](2) NULL,
	[ClientTypeCode] [varchar](7) NULL,
	[ClientTypeID] [int] NULL,
	[ClientTypeDesc] [varchar](20) NULL,
	[ClientCodeZip] [varchar](10) NULL,
	[TYPE][VARCHAR](5),
	[ServiceLine] [varchar](9) NULL,
	[TotalAdjustments] [money] NULL,
	[TotalCharges] [money] NULL,
	[TotalPayments] [money] NULL,
	[Variance] [varchar](25) NULL,
	[LocationCode] [varchar](10) NULL,
	[PatientRace] [varchar](30) NULL,
	[RegistrationLocation] [varchar](10) NULL,
	[Service] [varchar](40) NULL,
	[ServiceGroup] [varchar](22) NULL,
)
GO

CREATE TABLE [dbo].[AdjtTran_fact](
	[Id_col] [int] IDENTITY(1,1) NOT NULL,
	[ServiceNumber] [varchar](12) NULL,
	[EntityCode] [varchar](4) NULL,
	[EntityDesc] [varchar](40) NULL,
	[FacilityCode] [varchar](30) NULL,
	[Facility] [varchar](30) NULL,
	[TxnServiceDate] [date] NULL,
	[TxnServiceDateID] [int] NULL,
	[Amount] [money] NULL,
	[TransactionType] [varchar](10) NULL,
	[TxnCode] [varchar](15) NULL,
	[TxnCodeID] [int] NULL,
	[TxnCodeDesc] [varchar](80) NULL,
	[PostDate] [date] NULL,
	[PostDateID] [int] NULL
)
```

```
GO

CREATE TABLE [dbo].[Charges_fact](
      [ID_col] [int] IDENTITY(1,1) NOT NULL,
      [ServiceNumber] [varchar](12) NULL,
      [Amount] [money] NULL,
      [ChargeCode] [varchar](10) NULL,
      [ChargeCodeID] [int] NULL,
      [ChargeDepartment] [varchar](7) NULL,
      [ChargeDeptDesc] [varchar](30) NULL,
      [PostDate] [date] NULL,
      [PostDateID] [int] NULL,
      [Quantity] [smallint] NULL,
      [TxnServiceDate] [date] NULL,
      [TxnServiceDateID] [int] NULL,
      [FacilityCode] [varchar](4) NULL,
      [Facility] [varchar](30) NULL,
      [EntityCode] [varchar](4) NULL,
      [EntityDesc] [varchar](40) NULL,
      [UnitPrice] [money] NULL
)
GO

      CREATE TABLE [dbo].[DicDate](
            [DateID] [int] NULL,
            [Date] [date] NULL,
            [DayOfMonth] [tinyint] NULL,
            [MonthID] [int] NULL,
            [Month] [tinyint] NULL,
            [MonthName] [varchar](10) NULL,
            [MonthName_Short] [char](3) NULL,
            [WeekID] [int] NULL,
            [WeekOfMonth] [tinyint] NULL,
            [WeekOfYear] [tinyint] NULL,
            [DayOfYear] [smallint] NULL,
            [WeekdayID] [int] NULL,
            [WeekDay] [tinyint] NULL,
            [WeekdayName] [varchar](10) NULL,
            [WeekDayName_Short] [char](3) NULL,
            [Quarter] [tinyint] NULL,
            [QuarterName] [varchar](6) NULL,
            [Year] [int] NULL,
            [FiscalYear] [int] NULL,
            [IsWeekend] [bit] NULL
      )

      GO


with dates_cte(dt)
      as (select distinct t.dt from(
                  select cast(AdmitDate as date) as dt from Services_raw
                  union all
                  select cast(DischargeDate as date) as dt from Services_raw
                  union all
                  select cast(FinalBillDate as date) as dt from Services_raw
                  union all
                  select cast(TxnPostDate as date) as dt from AdjtTran_raw
                  union all
                  select cast(ChargePostDate as date) as dt from Charges_raw
                  union all
                  select cast(ChargeServiceDate as date) as dt from
Charges_raw)t
            where t.dt is not null
```

```
      )

      INSERT INTO [dbo].[DicDate]
                     ([DateID]
                     ,[Date]
                     ,[DayOfMonth]
                     ,[MonthID]
                     ,[Month]
                     ,[MonthName]
                     ,[MonthName_Short]
                     ,[WeekID]
                     ,[WeekOfMonth]
                     ,[WeekOfYear]
                     ,[DayOfYear]
                     ,[WeekdayID]
                     ,[WeekDay]
                     ,[WeekdayName]
                     ,[WeekDayName_Short]
                     ,[Quarter]
                     ,[QuarterName]
                     ,[Year]
                     ,[FiscalYear]
                     ,[IsWeekend])
      select
            [DateID] = YEAR(dt) * 10000 + MONTH(dt) * 100 + DAY(dt)
         ,[Date] = dt
         ,[DayOfMonth] = DAY(dt)
         ,[MonthID] = YEAR(dt) * 100 + MONTH(dt)
         ,[Month] = MONTH(dt)
         ,[MonthName] = DATENAME(mm, dt)
         ,[MonthName_Short] = UPPER(LEFT(DATENAME(mm, dt), 3))
         ,[WeekID] = YEAR(dt) * 100 + DATEPART(wk, dt)
         ,[WeekOfMonth] = DATEPART(WEEK, dt) - DATEPART(WEEK, DATEADD(MM,
DATEDIFF(MM, 0, dt), 0)) + 1
         ,[WeekOfYear] = DATEPART(wk, dt)
         ,[DayOfYear] = DATENAME(dy, dt)
         ,[WeekdayID] = YEAR(dt) * 1000 + DATEPART(wk, dt) *10 + DATEPART(dw,
dt)
         ,[WeekDay] = DATEPART(dw, dt)
         ,[WeekdayName] = DATENAME(dw,dt)
         ,[WeekDayName_Short] = UPPER(LEFT(DATENAME(dw, dt), 3))
         ,[Quarter] = DATEPART(q, dt)
         ,[QuarterName] = CASE WHEN DATENAME(qq, dt) = 1 THEN 'First'
                                          WHEN DATENAME(qq, dt) = 2 THEN
'second'
                                          WHEN DATENAME(qq, dt) = 3 THEN
'third'
                                          WHEN DATENAME(qq, dt) = 4 THEN
'fourth'
                                    END
         ,[Year] = YEAR(dt)
         ,[FiscalYear] = CASE WHEN MONTH(dt) >= 10 THEN YEAR(dt) + 1 ELSE
YEAR(dt)
                               END
         ,[IsWeekend] = CASE WHEN DATENAME(dw, dt) = 'Sunday'
                                          OR DATENAME(dw, dt) = 'Saturday' THEN
1 ELSE 0
                             END
         from dates_cte t
         left join [DicDate] dtcode
         on t.dt = dtcode.[Date]
         where dtcode.[DateID] is null ;

CREATE procedure [dbo].[sp_Services_stg_loading]
```

```
as
      truncate table [Services_stg]

      INSERT INTO [dbo].[Services_stg]
      SELECT srv.[ServiceNumber]
             ,DATEDIFF(day, cast(srv.DischargeDate as date), cast(l.FileDate as
date))
             ,0--ISNULL(ag.ageID, 0)
             ,cast(l.[FileDate] as date)
             ,ISNULL(YEAR(cast(l.[FileDate] as date)) * 10000 + MONTH(cast(l.
[FileDate] as date)) * 100 + DAY(cast(l.[FileDate] as date)), 19000101)
             ,srv.[AdmissionDxCode]
             ,ISNULL(dx.DxCodeID, 0)
             ,srv.[AdmissionDxCodeDesc]
             ,cast([AdmitDate]  as date)
             ,ISNULL(YEAR(cast(srv.[AdmitDate] as date)) * 10000 +
MONTH(cast(srv.[AdmitDate] as date)) * 100 + DAY(cast(srv.[AdmitDate] as date)),
19000101)
             ,srv.[AdmitSourceCode]
             ,srv.[AdmitSourceCodeDesc]
             ,srv.[AdmitTime]
             ,srv.[AdmitTypeCode]
             ,srv.[AdmitTypeCodeDesc]
             ,cast(srv.[CurrentBalance] as money)
             ,tr.[ThresholdID]
             ,srv.([DischargeDate] as date)
             ,ISNULL(YEAR(cast(srv.[DischargeDate] as date)) * 10000 +
MONTH(cast(srv.[DischargeDate] as date)) * 100 + DAY(cast(srv.[DischargeDate] as
date)), 19000101)
             ,srv.[DischargeDisposition]
             ,srv.[DischargeDispositionDesc]
             ,cast(srv.[DischargeTime] as time)
             ,srv.[EntityCode]
             ,srv.[EntityDesc]
             ,srv.[ERLevelCode]
             ,srv.[FacilityCode]
             ,srv.[FacilityDesc]
             ,cast([FinalBillDate] as date)
             ,ISNULL(YEAR(cast(srv.[FinalBillDate] as date)) * 10000 +
MONTH(cast(srv.[FinalBillDate] as date)) * 100 + DAY(cast(srv.[FinalBillDate] as
date)), 19000101)
             ,cast(srv.[InitialBillDate] as date)
             ,srv.[LengthofService(LOS)]
             ,srv.[Observation]
             ,srv.[ClientAge]
             ,srv.[ClientClassification]
             ,srv.[ClientFirstName]
             ,srv.[ClientLastName]
             ,srv.[ClientStatus]
             ,srv.[ClientType]
             ,ISNULL(pt.ClientTypeID, 0)
             ,srv.[ClientTypeDesc]
             ,srv.[ClientZip]
             ,srv.[TYPE]
             ,srv.[ServiceLine]
             ,cast(srv.[TotalAdjustments] as money)
             ,cast(srv.[TotalCharges] as money)
             ,cast(srv.[TotalPayments] as money)
             ,srv.[ProductLine]
             ,srv.[Location]
             ,srv.[RegistrationLocation]
             ,srv.[Service]
             ,srv.[ServiceGroup]
```

```
              FROM [dbo].[Services_raw] srv
        left join dbo.DicDxCode dx on srv.[AdmissionDxCode] = dx.DxCode
        left join  [dbo].[DicClientType] pt on srv.[ClientType] = pt.ClientType
        left join [dbo].[DicBalanceThreshold] tr on srv.CurrentBalance between
tr.MinValue and tr.MaxValue
        --left join [dbo].[AgingbyDsch] ag on DATEDIFF(day,
cast(srv.DischargeDate as date), cast(srv.PostDate as date)) between ag.ageMin
and ag.ageMax
        left join (select distinct ImportFileName, FileDate from [dbo].
[LogImportInfo]) l on srv.ImportFileName = l.ImportFileName
        insert into [dbo].[LogStgInfo]([TableNameSource], StgRowCount,
LoadingDate, SourceFileName)
        select top 1 [TableNameSource] = '[dbo].[Services_stg]',
                       [StgRowCount] = @@ROWCOUNT,
                     [LoadingDate] = getDate(),
                     [SourceFileName] = ImportFileName
        from [dbo].[LogImportInfo] where  ImportFileName like 'Services_%'


        update dbo.Services_stg
        set [AgingID] = ISNULL(ag.ageID, 0)
        from dbo.Services_stg srv
        left join [dbo].[AgingbyDsch] ag on DATEDIFF(day, cast(srv.DischargeDate
as date), cast(srv.PostDate as date)) between ag.ageMin and ag.ageMax
GO


      USE [test_of]

      CREATE TABLE [dbo].[LogFactInfo](
            [LogID] [int] IDENTITY(1,1) NOT NULL,
            [TableNameTarget] [varchar](50) NULL,
            [RowsAccumulated] [int] NULL,
            [NewRowsLoaded] [int] NULL,
            [LoadingDate] [datetime] default getDate()
      )
      GO
      CREATE TABLE [dbo].[LogStgInfo](
            [LogID] [int] IDENTITY(1,1) NOT NULL,
            [TableNameSource] [varchar](100) NULL,
            [StgRowCount] [int] NULL,
            [LoadingDate] [datetime] default getDate(),
            [SourceFileName] [varchar](100) NULL
      )
      GO

      CREATE TABLE [dbo].[LogImportInfo](
            [LogID] [int] IDENTITY(1,1) NOT NULL,
            [ImportFileName] [varchar](100) NULL,
            [FileRowCount] [int] NULL,
            [FileDate] [date] NULL
      )
      GO

----Reading data from files and loading into raw tables
exec [dbo].[sp_services_extract] '20200323'
exec [dbo].[sp_adjttran_extract] '20200323'
exec [dbo].[sp_charges_extract] '20200323'

exec [dbo].[dic_update] --Loading values into dimensional tables which exists in
source file and not in current dictionaries

--------Loading data from raw to staging tables. Mapping keys and doing explicit
conversion for date and money--------------------------------
```

```
exec [dbo].[sp_services_stg_loading]
exec [dbo].[sp_adjttran_stg_loading]
exec [dbo].[sp_charges_stg_loading]
------Loading data into fact tables-----
exec [dbo].[sp_services_fact_loading]
exec [dbo].[sp_adjttran_fact_loading]
exec [dbo].[sp_charges_fact_loading]
```

ДОДАТОК Б

Графічний матеріал атестаційної роботи

Харківський національний університет радіоелектроніки
Кафедра АПОТ

Система обліку та управління з використанням Fog-технології

Виконав:
Студент гр. СКСм-19-2 Федота О.В.

Керівник:
Проф. кафедри АПОТ Немченко В.П.

Харків 2021

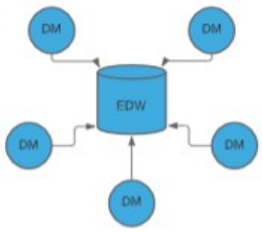
Мета та задачі роботи

Мета роботи: реалізувати компоненти системи обліку та управління на підприємстві

Задачі роботи:

- Визначити можливі області використання систем обліку та управління;
- Проаналізувати існуючі рішення і підходи до проектування аналогічних систем;
- Визначити вимоги до компонентів системи;
- Обрати відповідні варіанти реалізації ;
- Реалізувати окремі компоненти системи;
- Визначити ефективність використання спроектованої системи в рамках сучасного підприємства;

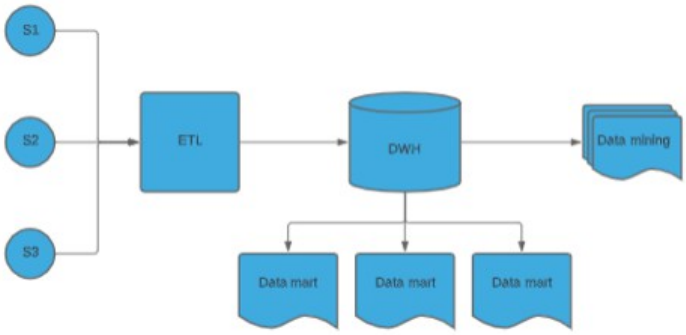
Поняття корпоративного сховища даних(EDW)



- EDW - це структуроване та централізоване місце, де користувачі можуть отримати доступ до бізнес-даних.
- EDW вирішує проблеми, пов'язані з елеваторами даних, шляхом передачі всіх відповідних даних підприємства у доступне центральне сховище для аналізу на всьому підприємстві.
- EDW забезпечує централізацію корпоративних даних, збільшену доступність цих даних для користувачів у різних бізнес-підрозділах та вдосконалення організації, структури та автоматизації зберігання та обробки даних.

3

Компоненти сучасного сховища даних



- Central database
- Data integration
- Data mart
- Metadata
- DWH access tools
- Data sources

4

Сучасні підходи до проектування сховищ даних

- Нормалізований підхід(3NF)
    1. легше отримати доступ до кваліфікованих ресурсів;
    2. високоструктурований;
    3. підходить для завантаження в режимі реального часу;
    4. легко зрозуміти та порівняно легко розширити його.
    5. інформація зберігається ефективно;
- Вимірне моделювання(Dimensional modelling)
    1. Дані упорядковані за предметною областю;
    2. Легко зрозумілий для бізнес-користувачів;
    3. Мінімізує реплікацію даних;
    4. Підходить для історичного зберігання;
    5. Легко зв'язати точки агрегування;
- Гібридне моделювання

5

Таблиці фактів і виміри

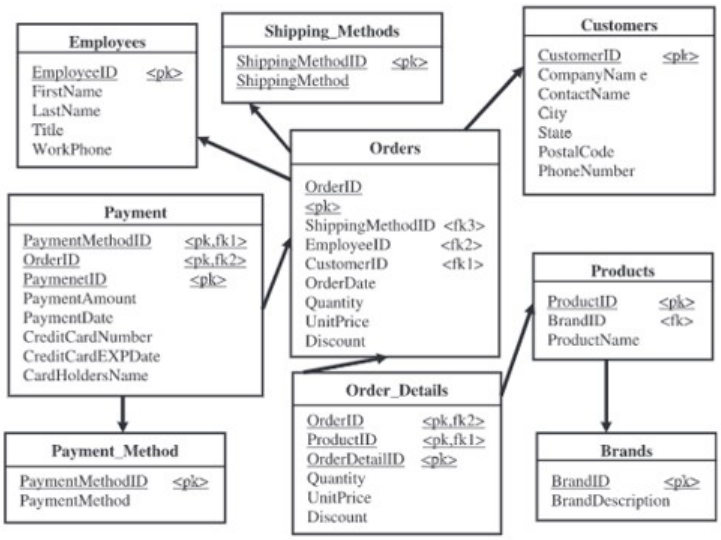
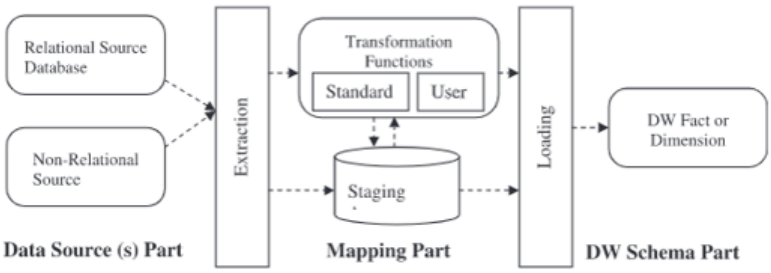


Схема зв'язку таблиць вимірів і фактів

6

Організація ETL-процесів у корпоративному сховищі даних



Система Extract-Transform-Load (ETL) є основою сховища даних. Правильно розроблена система ETL витягує дані з вихідних систем, забезпечує дотримання стандартів якості та узгодженості даних, узгоджує дані, щоб окремі джерела могли використовуватися разом, і нарешті доставляє дані у готовому до відображення форматі, щоб розробники програм могли створювати додатки та кінцевих користувачів може приймати рішення.

Система ETL додає значну цінність даним. Це набагато більше, ніж інструмент для виведення даних із вихідних систем у сховище даних. Зокрема, система ETL:

1. усуває помилки та виправляє відсутні дані;
2. налаштовує дані з кількох джерел для спільного використання;
3. структурує дані, які можуть бути використані інструментами кінцевих користувачів;

7

Етап вилучення даних

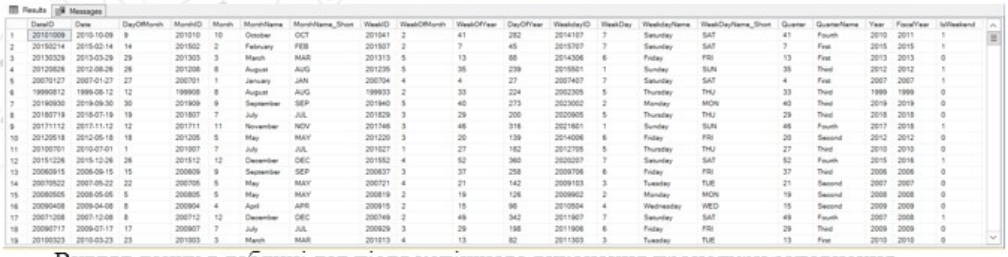
Після визначення вимог наступним питанням є проектування таблиць, які будуть первинним місце призначення для даних, що завантажуються з зовнішніх файлів в систему. Для запобігання помилок під час процесу та швидкої ідентифікації проблем на першому етапі необхідно завантажувати дані в тому вигляді, в якому вони представлені в джерелі не виконуючи жодних явних або неявних перетворень.

```
CREATE procedure [dbo].[sp_services_extract]
        @dt varchar(8)
as
        truncate table Services_raw
        declare @services_raw varchar(max)
        set @services_raw = '
        bulk insert dbo.Services_raw
        from "C:\Users\Oleks\Flat File  Parsing\Services_' + @dt + '.txt" ' + '
        with(FORMATFILE='"C:\Users\Oleks\Flat File  Parsing\Format Files\Services.fmt"',
         firstrow = 2)'
        exec(@accounts_raw)

        insert into LogImportInfo(ImportFileName, FileDate, FileRowCount)
        select  ImportFileName = 'Services_' + @dt + '.txt'
                  ,FileDate =  DATEADD(day, -1, cast(@dt as date))
                  ,FileRowCount = @@ROWCOUNT

        update dbo.Services_raw set [ImportFileName] = 'Services_' + @dt + '.txt'
GO
```

Для цього використовуються таблиці, що призначені для збереження сирих даних. Процедури повинні отримувати дату в імені файла як параметр та парсити джерело за конкретним місцезнаходженням в файловій системі. Наведено приклад процедури парсингу файлу та завантаження в таблицю логів для таблиці Services_raw. Аналогічні процедури створені для завантаження таблиць Charges_raw та AdjtTran_raw.

8

Поширені види таблиць вимірів



	DateID	Date	DayOfMonth	MonthID	Month	MonthName	MonthName_Short	WeekID	WeekOfMonth	WeekOfYear	DayOfYear	WeekdayID	WeekDay	WeekdayName	WeekDayName_Short	Quarter	QuarterName	Year	FiscalYear	IsWeekend
1	20101009	2010-10-09	9	201010	10	October	OCT	201041	2	41	282	2014107	7	Saturday	SAT	41	Fourth	2010	2011	1
2	20150214	2015-02-14	14	201502	2	February	FEB	201507	2	7	45	2015707	7	Saturday	SAT	7	First	2015	2015	1
3	20130329	2013-03-29	29	201303	3	March	MAR	201313	5	13	88	2014306	6	Friday	FRI	13	First	2013	2013	0
4	20120826	2012-08-26	26	201208	8	August	AUG	201235	5	35	239	2015501	1	Sunday	SUN	35	Third	2012	2012	1
5	20070127	2007-01-27	27	200701	1	January	JAN	200704	4	4	27	2007407	7	Saturday	SAT	4	First	2007	2007	1
6	19990812	1999-08-12	12	199908	8	August	AUG	199933	2	33	224	2002305	5	Thursday	THU	33	Third	1999	1999	0
7	20190930	2019-09-30	30	201909	9	September	SEP	201940	5	40	273	2023002	2	Monday	MON	40	Third	2019	2019	0
8	20180719	2018-07-19	19	201807	7	July	JUL	201829	3	29	200	2020905	5	Thursday	THU	29	Third	2018	2018	0
9	20171112	2017-11-12	12	201711	11	November	NOV	201746	3	46	316	2021601	1	Sunday	SUN	46	Fourth	2017	2018	1
10	20120518	2012-05-18	18	201205	5	May	MAY	201220	3	20	139	2014006	6	Friday	FRI	20	Second	2012	2012	0
11	20100701	2010-07-01	1	201007	7	July	JUL	201027	1	27	182	2012705	5	Thursday	THU	27	Third	2010	2010	0
12	20151226	2015-12-26	26	201512	12	December	DEC	201552	4	52	360	2020207	7	Saturday	SAT	52	Fourth	2015	2016	1
13	20060915	2006-09-15	15	200609	9	September	SEP	200637	3	37	258	2009706	6	Friday	FRI	37	Third	2006	2006	0
14	20070522	2007-05-22	22	200705	5	May	MAY	200721	4	21	142	2009103	3	Tuesday	TUE	21	Second	2007	2007	0
15	20080505	2008-05-05	5	200805	5	May	MAY	200819	2	19	126	2009902	2	Monday	MON	19	Second	2008	2008	0
16	20090408	2009-04-08	8	200904	4	April	APR	200915	2	15	98	2010504	4	Wednesday	WED	15	Second	2009	2009	0
17	20071208	2007-12-08	8	200712	12	December	DEC	200749	2	49	342	2011907	7	Saturday	SAT	49	Fourth	2007	2008	1
18	20090717	2009-07-17	17	200907	7	July	JUL	200929	3	29	198	2011906	6	Friday	FRI	29	Third	2009	2009	0
19	20100323	2010-03-23	23	201003	3	March	MAR	201013	4	13	82	2011303	3	Tuesday	TUE	13	First	2010	2010	0

Вигляд даних в таблиці дат після успішного виконання процедури заповнення

```
INSERT INTO [dbo].[DicDate]
select
      [DateID] = YEAR(dt) * 10000 + MONTH(dt) * 100 + DAY(dt)
    ,[Date] = dt
    ,[DayOfMonth] = DAY(dt)
    ,[MonthID] = YEAR(dt) * 100 + MONTH(dt)
    ,[Month] = MONTH(dt)
    ,[MonthName] = DATENAME(mm, dt)
    ,[MonthName_Short] = UPPER(LEFT(DATENAME(mm, dt), 3))
    ,[WeekID] = YEAR(dt) * 100 + DATEPART(wk, dt)
    ,[WeekOfMonth] = DATEPART(WEEK, dt) - DATEPART(WEEK, DATEADD(MM, DATEDIFF(MM, 0, dt), 0)) + 1
    ,[WeekOfYear] = DATEPART(wk, dt)
    ,[DayOfYear] = DATENAME(dy, dt)
    ,[WeekdayID] = YEAR(dt) * 1000 + DATEPART(wk, dt) *10 + DATEPART(dw, dt)
    ,[WeekDay] = DATEPART(dw, dt)
    ,[WeekdayName] = DATENAME(dw,dt)
    ,[WeekDayName_Short] = UPPER(LEFT(DATENAME(dw, dt), 3))
    ,[Quarter] = DATEPART(q, dt)
    ,[QuarterName] = CASE WHEN DATENAME(qq, dt) = 1 THEN 'First'
                          WHEN DATENAME(qq, dt) = 2 THEN 'second'
                          WHEN DATENAME(qq, dt) = 3 THEN 'third'
                          WHEN DATENAME(qq, dt) = 4 THEN 'fourth'
                     END
    ,[Year] = YEAR(dt)
    ,[FiscalYear] = CASE WHEN MONTH(dt) >= 10 THEN YEAR(dt) + 1 ELSE YEAR(dt)
                         END
    ,[IsWeekend] = CASE WHEN DATENAME(dw, dt) = 'Sunday'
                          OR DATENAME(dw, dt) = 'Saturday' THEN 1 ELSE 0
                        END
from dates_cte t
left join [DicDate] dtcode
on t.dt = dtcode.[Date]
where dtcode.[DateID] is null ;
```

```
with dates_cte(dt)
      as (select distinct t.dt from(
                  select cast(AdmitDate as date) as dt from Services_raw
                  union all
                  select cast(DischargeDate as date) as dt from Services_raw
                  union all
                  select cast(FinalBillDate as date) as dt from Services_raw
                  union all
                  select cast(TxnPostDate as date) as dt from AdjtTran_raw
                  union all
                  select cast(ChargePostDate as date) as dt from Charges_raw
                  union all
                  select cast(ChargeServiceDate as date) as dt from Charges_raw)t
            where t.dt is not null
      )
```

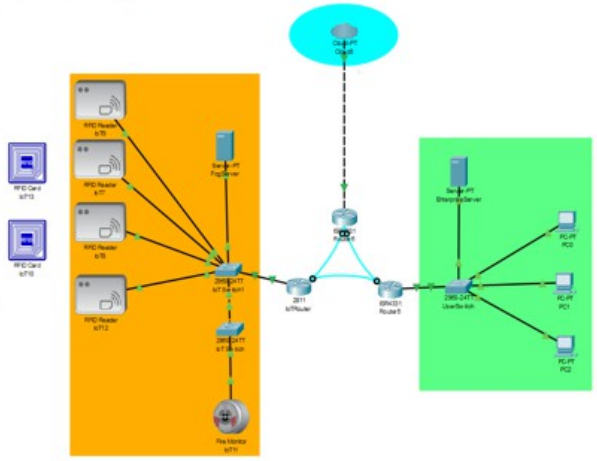
Фрагменти процедури заповнення таблиці виміру DicDate

Трансформація і завантаження в стейджингові таблиці

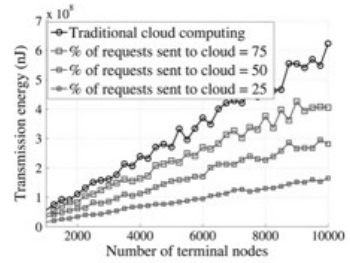
```
INSERT INTO [dbo].[Services_stg]
SELECT acc.[ServiceNumber]
          ,DATEDIFF(day, cast(acc.DischargeDate as date), cast(l.FileDate as date))
          ,0--ISNULL(ag.ageID, 0)
          ,cast(l.[FileDate] as date)
          ,ISNULL(YEAR(cast(l.[FileDate] as date)) * 10000 + MONTH(cast(l.[FileDate] as date)) * 100 + DAY(cast(l.[FileDate] as date)), 19000101)
          ,srv.[AdmissionDxCode]
          ,ISNULL(dx.DxCodeID, 0)
          ,cast([AdmitDate]  as date)
          ,ISNULL(YEAR(cast(srv.[AdmitDate] as date)) * 10000 + MONTH(cast(srv.[AdmitDate] as date)) * 100 + DAY(cast(srv.[AdmitDate] as date)), 19000101)
          ----------------------------------------------------------------------------------------------
          ,cast(srv.[CurrentBalance] as money)
          ,tr.[ThresholdID]
          ,srv.([DischargeDate] as date)
          ,ISNULL(YEAR(cast(srv.[DischargeDate] as date)) * 10000 + MONTH(cast(srv.[DischargeDate] as date)) * 100 + DAY(cast(srv.[DischargeDate] as date)), 19000101)
          ,srv.[DischargeDisposition]
          ,srv.[DischargeDispositionDesc]
          ,cast(srv.[DischargeTime] as time)
          ----------------------------------------------------------------------------------------------
          ,cast([FinalBillDate] as date)
          ,ISNULL(YEAR(cast(srv.[FinalBillDate] as date)) * 10000 + MONTH(cast(srv.[FinalBillDate] as date)) * 100 + DAY(cast(srv.[FinalBillDate] as date)), 19000101)
          ,cast(srv.[InitialBillDate] as date)
          ----------------------------------------------------------------------------------------------
          FROM [dbo].[Services_raw] srv
    left join dbo.DicDxCode dx on srv.[AdmissionDxCode] = dx.DxCode
    left join  [dbo].[DicClientType] pt on srv.[ClientType] = pt.ClientType
    left join [dbo].[DicBalanceThreshold] tr on srv.CurrentBalance between tr.Minvalue and tr.Maxvalue
    --left join [dbo].[AgingByDsch] ag on DATEDIFF(day, cast(srv.DischargeDate as date), cast(srv.PostDate as date)) between ag.ageMin and ag.ageMax
    left join (select distinct ImportFileName, FileDate from [dbo].[LogImportInfo]) l on srv.ImportFileName = l.ImportFileName
    insert into [dbo].[LogStgInfo]([TableNameSource], StgRowCount, LoadingDate, SourceFileName)
    select top 1 [TableNameSource] = '[dbo].[Services_stg]',
                 [StgRowCount] = @@ROWCOUNT,
                [LoadingDate] = getDate(),
                [SourceFileName] = ImportFileName
    from [dbo].[LogImportInfo] where  ImportFileName like 'Services_%'


    update dbo.Services_stg
    set [AgingID] = ISNULL(ag.ageID, 0)
    from dbo.Services_stg srv
    left join [dbo].[AgingbyDsch] ag on DATEDIFF(day, cast(srv.DischargeDate as date), cast(srv.PostDate as date)) between ag.ageMin and ag.ageMax
```

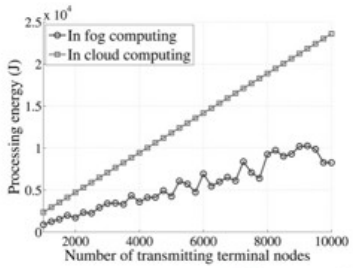
Реалізація процедури заповнення стейджингової таблиці

Дослідження ефективності Fog-обчислень



Модель з Fog-підсистемою в середовищі Cisco Packet Tracer





12

Завантаження в таблиці фактів

```
CREATE procedure [dbo].[sp_Services_fact_loading]
as
        declare @rcnt int

        delete FROM Services_fact
        WHERE ServiceNumber IN(select ServiceNumber From Services_stg)
        and year(PostDate) *  100 + month(PostDate) IN(select year(PostDate)
                        *  100 + month(PostDate) From Services_stg)

        update Services_fact
        set PostDate = (Select top 1 PostDate from Services_stg)
        where year(PostDate) *  100 + month(PostDate)
              IN(select year(PostDate) *  100 + month(PostDate) From Services_stg)

        set @rcnt = (select count(*) from Services_fact)

---Скрипт не включає частину Insert into і Select from….

        Insert Into LogFactInfo (TableNameTarget,RowsAccumulated,NewRowsLoaded,LoadingDate)
        Select TableNameTarget = 'Services_fact',
                  RowsAccumulated = @rcnt,
                  NewRowsLoaded = @@ROWCOUNT,
                  LoadingDate = GetDate()
GO
```

Фрагмент процедури завантаження таблиці фактів що реалізує логіку завантаження

```
----Reading data from files and loading into raw tables
exec [dbo].[sp_services_extract] '20200323'
exec [dbo].[sp_adjttran_extract] '20200323'
exec [dbo].[sp_charges_extract] '20200323'

exec [dbo].[dic_update] --Loading values into dimensional tables which exists in source file and not in current dictionaries

--------Loading data from raw to staging tables. Mapping keys and doing explicit conversion for date and money----------
exec [dbo].[sp_services_stg_loading]
exec [dbo].[sp_adjttran_stg_loading]
exec [dbo].[sp_charges_stg_loading]
-------Loading data into fact tables-----
exec [dbo].[sp_services_fact_loading]
exec [dbo].[sp_adjttran_fact_loading]
exec [dbo].[sp_charges_fact_loading]
```

```
/****** Script for SelectTopNRows command from SSMS
SELECT TOP (1000) [LogID]
      ,[TableNameTarget]
      ,[RowsAccumulated]
      ,[NewRowsLoaded]
      ,[LoadingDate]
  FROM [test_of].[dbo].[LogFactInfo]
  where TableNameTarget = 'Services_fact'
  order by LoadingDate desc
```

102 %

Results | Messages

	LogID	TableNameTarget	RowsAccumulated	NewRowsLoaded	LoadingDate
1	106	Services_fact	348118	350651	2021-05-11 05:51:59.543
2	103	Services_fact	957	347161	2021-05-11 05:49:36.487
3	100	Services_fact	0	346979	2021-05-11 05:47:04.750

Результат перевірки таблиці логів після завантаження трьох датасетів

11

Висновки

В рамках дослідницької роботи були розглянуті поняття Туманних обчислень, Сховищ даних, можливості взаємодії пристроїв "Інтернету речей" з сучасними системами обліку та управління. Були досліджені існуючі проблеми, що виникають під час проектування корпоративних сховищ даних, шляхи їх вирішення, а також актуальні підходи до створення аналітичних систем і систем збору обробки і завантаження даних.

- Спроектована і реалізована система, що представляє собою корпоративне сховище даних;
- в рамках реалізації КСД проаналізовані вимоги і виконане моделювання процесу ETL;
- виконано моделювання системи з використанням Fog-обчислень;
- були виконані перевірки роботоспроможності системи, що підтвердило повну роботоспроможність сховища даних і можливість його використання в подальшому;

Були запропоновані напрями розвитку створеної системи для забезпечення повної відповідності світовим стандартам подібних систем. Розглянуті аспекти покращення та розширення функціоналу системи, а саме: безпеки, гнучкості, масштабування та швидкості роботи.

Дякую за увагу!

Додаток В

Наукова публікація

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ
УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

МАТЕРІАЛИ
XXIV МІЖНАРОДНОГО МОЛОДІЖНОГО ФОРУМУ

**«РАДІОЕЛЕКТРОНІКА ТА МОЛОДЬ
У XXI СТОЛІТТІ»**

7 – 9 квітня 2020 р.

Том 5

КОНФЕРЕНЦІЯ
«ВІРТУАЛЬНИЙ ТА ФІЗИЧНИЙ КОМП'ЮТИНГ»

Харків 2020

АНАЛІЗ ЕФЕКТИВНОСТІ АНТИКОЛІЗІЙНИХ МЕХАНІЗМІВ В RFID-МЕРЕЖАХ

Федота О.В.
Науковий керівник – проф. Немченко В.П.
Харківський національний університет радіоелектроніки
(61166, Харків, пр. Науки,14, каф. АПОТ, тел. (057) 702-13-26)
e-mail:oleksandr.fedota@nure.ua, тел. 067- 455-455-0

In the present work, an analysis of the effectiveness of modern conflict resolution mechanisms in systems created using the Internet of things technologies such as RFID have been proposed. The object of research is the system of accounting and management based on radio frequency identification technology.

Вступ. Мережа RFID складається з трьох компонентів: 1) теги; 2) зчитувачі; 3) сервери / комп'ютери. Теги - це пристрої, які прикріплені до елементів, які слід ідентифікувати та відстежувати. Мітки містять інформацію, пов'язану з таким елементом, як виробник і ціна. Вони складається з компонентів зв'язку і мікросхем. Зчитувач може ідентифікувати об'єкти з вставкою тегів. На основі енергопостачання мітки включають активну мітку, пасивну мітку і напівактивну мітку. Пасивна мітка не має батареї і витягує енергію від зчитувача. Завдяки низькій потужності, низькій вартості і високій надійності, пасивна мітка широко використовується в процедурі оформлення потоку продукції. Зчитувачі, ідентифікують теги і збирають інформацію, що зберігається в мітках через радіочастотну зв'язок. Зчитувач і тег досягають один одного з одним стрибком, усуваючи необхідність користування мережевим шаром. Це виключає необхідність використовування транспортного рівня в стеку протоколів RFID. Мета дослідження – аналіз перспектив підвищення ефективности обробки і передачі даних в RFID системах і розробка проекту по впровадженню IoT-технологій для автоматизації складської логістики на підприємстві. Задача – розробка автоматичної системи ідентифікації з використанням IoT- технологій, дослідження існуючих проблем під час ідентифікації, та визначення методів їх вирішення.

Склад дослідження. Існують два типи протоколів рівня RFID MAC, детерміновані і ймовірнісні. Найпоширенішою детермінованою реалізацією є бінарне дерево. Практично всі поточні багатобічні реалізації засновані на протоколі ALOHA з-за його простоти.

Дискретний протокол ALOHA вимагає, щоб всі вузли синхронізували час початку передачі кадрів. У чистому протоколі ALOHA, коли прибуває перший кадр (тобто дейтаграма мережевого рівня передається на більш низький рівень передавального вузла), вузол негайно передає весь кадр цілком в широкомовний канал. Якщо переданий кадр стикається з одним або декількома іншими кадрами, з деякою ймовірністю вузол негайно

передає кадр повторно. В іншому випадку вузол вичікує протягом часу, необхідного для передачі одного кадру, після чого знову з тією ж ймовірністю передає кадр або перечікує ще один інтервал часу. Основною проблемою, пов'язаною з виконанням завдань RFID, є ефективність ідентифікації міток і швидкість ідентифікації. Існує кілька факторів, які впливають на продуктивність RFID. Оскільки основна мета полягає в прагненні моделювання продуктивності та імітації в середовищі OPNET, фактори обмежуються Q і обраним тегом, припускаючи, що Q не змінюється під час проведення інвентаризації. У протоколі повітряного інтерфейсу Gen 2, Q є параметром, який використовується для регулювання ймовірності відповіді мітки. Зчитувач вказує теги в заданому порядку, щоб вибрати випадкове число між нулем і 2 ^ Q; тег може бути успішно скомпонований, якщо інші теги не вибирають однакове випадкове число ("зіткнення"). Більш великі значення Q зменшують ймовірність зіткнення, але вимагають від читача більше часу під час сеансу. Ефективність ідентифікації міток визначається як наступне:

$$R = \frac{n_i}{n_a}, \quad (1)$$

де R позначає ефективність ідентифікації під час зчитування, n_i позначає кількість ідентифікованих міток, а n означає кількість спроб тегів. Завдяки зіткненню міток, R зазвичай є меншою, ніж швидкість ідентифікації тегів, яка тісно пов'язана з використанням каналу. Він вимірює кількість тегів, які ідентифікуються в межах одиниці часу. Вона визначається як наступне:

$$S = \frac{n_i}{T}, \quad (2)$$

де S позначає швидкість ідентифікації, а T означає час, що використовується в секундах для ідентифікації цієї групи тегів. n_i є таким же, як у (1). Результати отримані шляхом виконання серії моделювань в пакетному режимі з кожним циклом, що змінюють значення параметра моделі Q.

Висновки. Наукова новизна визначається використанням найбільш ефективного механізму взаємодії між системою обліку та управління, RFID-тегами і зчитувачами, а також впровадженням найбільш актуальних алгоритмів передачі даних, які б могли вирішувати існуючі проблеми, що в результаті дозволить використовувати на підприємстві найбільш економічно вигідну і сучасну систему обліку.

Список використаних джерел:

1. K. Finkenzeller, RFID-technologies. Reference guide. 2016. P. 211-223.

2. Зіборов І. А. Застосування RFID технологій в діяльності різних суб'єктів господарювання // Молодий вчений. - 2009. С. 11-26.

3. K. Finkenzeller, RFID handbook – 2nd Ed., John Wiley & Sons, 2003. P. 145-151.

Додаток Г

ФОРМА ВІДОМОСТІ МАГІСТЕРСЬКОЇ АТЕСТАЦІЙНОЇ РОБОТИ

Номер докумен-та	Познака	*Найменування*	*Прим.*
		Текстові документи	
1	ГЮІК. ХХХХХХ.005 ПЗ	Пояснювальна записка	83с.
		Графічні документи	
2		Слайд-перезентація	14 слайдів
		Носії інформації	
3		Компакт-диск(CD)	1 шт

Зм	*Арк*	*№ докум.*	*Підпис*	*Дата*	ХХХХХХ.005 ПЗ		
Розроб.		Федота О.В.			Система обліку та управління підприємтсвом з використанням Fog-технологій	*Літ*	*Арк*	*Аркуш*
Перевір.		Немченко В.П.						
Н.контр		Немченко В.П.				***ХНУРЕ***		
Затв.		Чумаченко С.В.				*Кафедра АПОТ*		