ISEL | DEETC

Sistemas de Informação 2

Tantras Enamour (Parte I e II)

Semestre de Verão de 2010/2011

Professor: Porfírio Filipe
Autores:
31401 – Nuno Cancelo
32142 – Cláudia Crisóstomo
33595 – Nuno Sousa
Turma: LI51N

Índice

Introdução

A elaboração deste projeto tende a cumprir os objetivos propostos no enunciado. Esses objetivos apesar de identificados surgem como pontos de aprendizagem académica interessantes que de um forma ou outra não foi possível aprofundar no contexto de outras cadeiras.

A análise do enunciado foi efetuada em três fases:

- descrição das entidades e relações explicitamente indicadas
- descrição de entidades e relações que nos sugeriam um boa aproximação
- inclusão de entidade e relações devido a necessidades de projeção de implementação

Pretende-se ao longo do relatório explicar a forma e as decisões tomadas na implementação do nosso desenho da solução.

Parte 1

Modelo de Dados Conceptual

Após a analise do enunciado obtivemos este esboço do modelo conceptual da nossa solução na Figura 1: Esboço do Modelo Conceptual – Parte 1

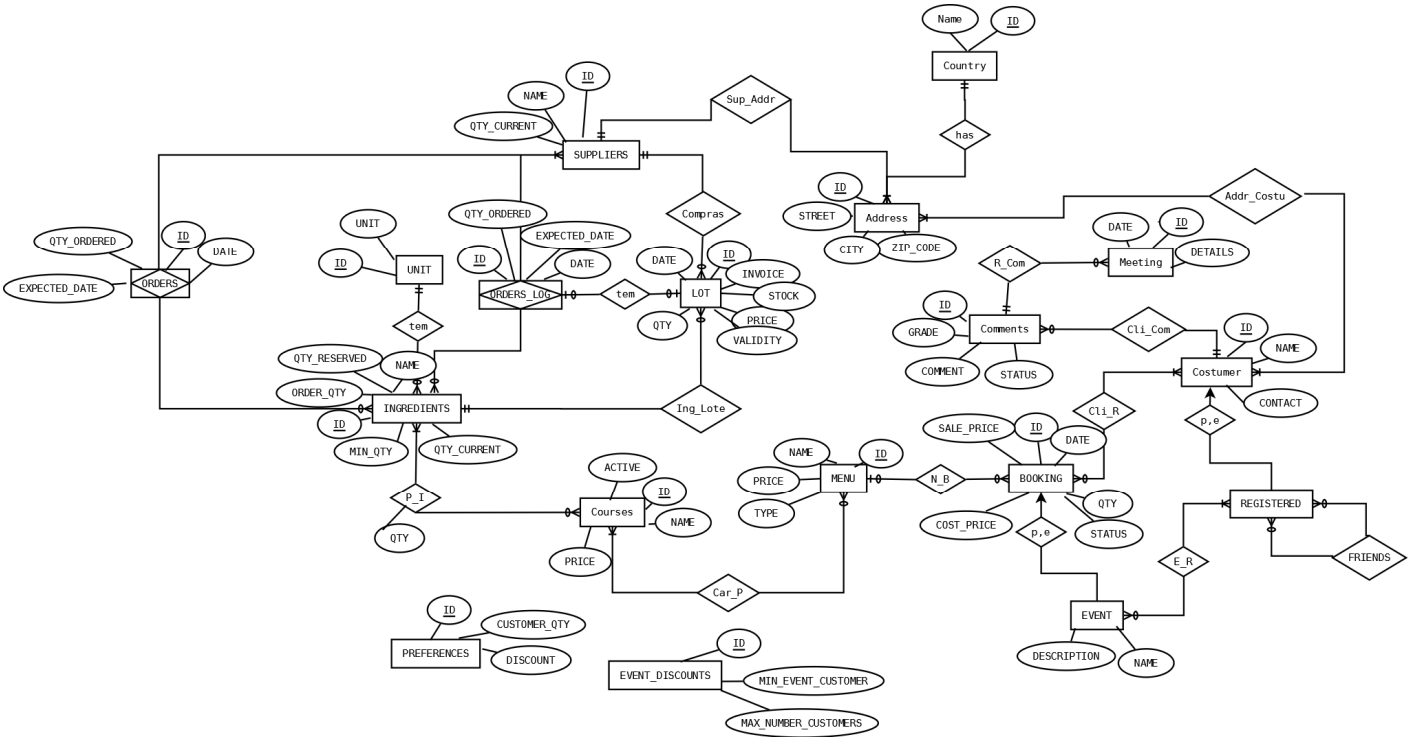

Figura 1: Esboço do Modelo Conceptual – Parte 1

A descrição pormenorizada de cada entidade e relação será indicada na secção do modelo lógico.

Modelo de Dados Lógico

Nesta secção apresentamos o modelo lógico da representação Entidade Associação anterior. Como forma de internacionalização da solução e para evitar erros de caracteres inválidos nos nomes das tabelas e dos campos, optámos por renomear os campos do Modelo EA para os correspondentes em Inglês.

Na descrição de cada Entidade e relação é efectuada a relação entre modelos.

Entidade Cliente:

Entidade que contem a identificação de um cliente.

- CUSTOMER(ID[PK],NAME,CONTACT,TYPE);
  - ID [PK]
    - INTEIRO
  - NAME
    - STRING
  - CONTACT
    - INTEIRO
  - TYPE
    - INTEIRO
    - Valores Admitidos {0,1}

Chaves Candidatas={ID}

Entidade Registrado:

Como extensão de Cliente, um cliente registado contêm mais alguns atributos que os diferenciam dos demais. No entanto tem mais uma característica imposta pelo enunciado ("Só são aceites reservas de clientes registados em relação aos quais se mantém informação sobre: listas de amigos (que também são clientes)...").

- REGISTERED(CUSTOMER_ID[FK], ADDRESS_ID[FK]);
  - CUSTOMER_ID [PK] [FK]
    - INTEIRO
  - ADDRESS_ID[FK]
    - INTEIRO

Chaves Candidatas={CUSTOMER_ID}
Chaves Estrangeiras={CUSTOMER_ID},{ADDRESS_ID}

Entidade Prato:

Identificação de um prato

- COURSES(ID[PK], NAME, ACTIVE, VALUE);
  - ID[PK]
    - INTEIRO
  - NAME
    - STRING
  - ACTIVE
    - BIT
  - PRICE
    - DECIMAL
  - TYPE
    - CHAR (20)
    - NOT NULL

Chaves Candidatas={ID}

Entidade Comentário:

Descrição de um comentário

- COMMENTS(ID[PK], CUSTOMER_ID[FK], COURSES_ID[FK], GRADE, COMMENT);
  - ID[PK]
    - INTEIRO
  - CUSTOMER_ID[FK]
    - INTEIRO
  - COURSES_ID[FK]
    - INTEIRO
  - GRADE,
    - INTEIRO
    - [1,5]
  - COMMENT
    - STRING

Chaves Candidatas={ID}
Chaves Estrangeiras={CUSTOMER_ID},{COURSES_ID}

Entidade Reunião:

Dados relativos a uma reunião originada por um comentário negativo

- MEETING(ID[PK],COMMENT_ID[FK], DATE,DETAILS);
  - ID[PK]
    - INTEIRO
  - COMMENT_ID[FK]
    - INTEIRO
  - DATE
    - DATA
  - DETAILS
    - STRING

Chaves Candidatas={ID}
Chaves Estrangeiras={COMMENT_ID}

Entidade Cardápio:

Descrição de um Cardápio

- MENU(ID[PK], NAME, TYPE);
  - ID[PK]
    - STRING
  - NAME
    - STRING
  - TYPE
    - STRING

Chaves Candidatas={ID}

Entidade Reserva:

Descrição de uma reserva

- BOOKING(ID[PK], CUSTOMER_ID[FK], DATE, QTY,TYPE,STATUS);
  - ID[PK]
    - INTEIRO
  - CUSTOMER_ID[FK]
    - INTEIRO
  - DATE
    - DATA
  - QTY

        - INTEIRO
    - TYPE
        - INTEIRO
        - {0,1} (0 - Normal Booking; 1 - Event)
    - STATUS
        - INTEIRO
        - [0,2] (0 - PENDING; 1 - CONFIRMED; 2 - CANCELED)

Chaves Candidatas={ID}
Chaves Estrangeiras={CUSTOMER_ID}

Entidade Evento:

Descrição de um evento gastronómico

- EVENT(BOOKING_ID[FK], MENU_ID[FK], NAME, DESCRIPTION);
    - BOOKING_ID[PK][FK]
        - INTEIRO
    - MENU_ID[FK]
        - INTEIRO
    - NAME
        - STRING
    - DESCRIPTION
        - STRING

Chaves Candidatas={BOOKING_ID}
Chaves Estrangeiras={BOOKING_ID},{MENU_ID}

Entidade Normal:

Descrição de um evento genérico como seja uma reserva individual.

- NORMAL_BOOKING(BOOKING_ID[FK], MENU_ID[FK]);
    - BOOKING_ID[PK][FK]
        - INTEIRO
    - MENU_ID[FK]
        - INTEIRO

//pode ter ou não menu escolhido à partida

Chaves Candidatas={BOOKING_ID}
Chaves Estrangeiras={BOOKING_ID},{MENU_ID}

Entidade Unidade:

Descrição dos vários tipos de unidades que se admitem válidas, relacionadas com a confeção de pratos

- UNIT(ID[PK], UNIT);
    - ID[PK]
        - INTEIRO
    - UNIT
        - STRING

Chaves Candidatas={ID}

Entidade Ingrediente:

Descrição de um ingrediente

- INGREDIENTS(ID[PK], NAME, QTY_RESERVED, UNIT_ID[FK]);
    - ID[PK]
        - INTEIRO
    - NAME
        - STRING
    - QTY_RESERVED

        - FLOAT
        - DEFAULT 0
    - QTY_CURRENT
        - DECIMAL (10,3)
        - DEFAULT 0
        - NOT NULL
    - UNIT_ID[FK]
        - INTEIRO

Chaves Candidatas={ID}
Chaves Estrangeiras={UNIT_ID}

Entidade Fornecedor:

Descrição de um fornecedor

- SUPPLIERS(ID[PK], NAME, ADDRESS_ID[FK]);
    - ID[PK]
        - INTEIRO
    - NAME
        - STRING
    - ADDRESS_ID[FK]
        - INTEIRO

Chaves Candidatas={ID}
Chaves Estrangeiras={ADDRESS_ID}

Entidade Lote:

Descrição para compras em Lotes

- LOT(ID[PK], INGREDIENTS_ID[FK], SUPPLIER_ID[FK], INVOICE, DATE, QTY, PRICE, VALIDITY, STOCK);
    - ID[PK]
        - INTEIRO
    - INGREDIENTS_ID[FK]
        - INTEIRO
    - SUPPLIER_ID[FK]
        - INTEIRO
    - INVOICE
        - INTEIRO
    - DATE
        - DATA
    - QTY
        - DECIMAL
        - DEFAULT 0
    - PRICE
        - DECIMAL
        - DEFAULT 0
    - VALIDITY
        - DATA
    - STOCK
        - DECIMAL
        - DEFAULT 0

Chaves Candidatas={ID}
Chaves Estrangeiras={INGREDIENTS_ID}{SUPPLIER_ID}

Entidade Encomendas:

Descrição de um encomenda

- ORDERS(ID[PK],SUPPLIER_ID[FK], INGREDIENT_ID[FK], DATE, QTY_ORDERED, EXPECTED_DATE);
    - ID[PK]

        - INTEIRO
    - SUPPLIER_ID[FK]
        - INTEIRO
    - INGREDIENT_ID[FK]
        - INTEIRO
    - DATE
        - DATA
    - QTY_ORDERED
        - INTEIRO
    - EXPECTED_DATE
        - DATA

Chaves Candidatas={ID}
Chaves Estrangeiras={SUPPLIER_ID}{INGREDIENT_ID}

Entidade Encomendas_Log:

Mantém o histórico de encomendas.

- ORDERS_LOG(ID[PK],SUPPLIER_ID[FK], INGREDIENT_ID[FK], DATE, QTY_ORDERED, EXPECTED_DATE, LOT_ID[FK]);
    - ID[PK]
        - INTEIRO
    - SUPPLIER_ID[FK]
        - INTEIRO
    - INGREDIENT_ID[FK]
        - INTEIRO
    - DATE
        - DATA
    - QTY_ORDERED
        - INTEIRO
    - EXPECTED_DATE
        - DATA

Chaves Candidatas={ID}
Chaves Estrangeiras={SUPPLIER_ID}{INGREDIENT_ID}

Entidade Preferencias:

Descrição de moradas

- PREFERENCES(ID[PK], MIN_EVENT_CUSTOMER, MAX_NUMBER_CUSTOMERS);
    - ID[PK]
        - INTEIRO
    - MIN_EVENT_CUSTOMER
        - INTEIRO
        - DEFAULT 0
    - MAX_NUMBER_CUSTOMERS
        - INTEIRO
        - DEFAULT 100

Chaves Candidatas={ID}

Entidade Eventos_Discontos:

Descrição de moradas

- EVENT_DISCOUNTS(ID[PK], CUSTOMER_QTY, CITY,DISCOUNT);
    - ID[PK]
        - INTEIRO
    - CUSTOMER_QTY
        - INTEIRO
        - >= 0
        - DEFAULT 0
    - CITY
        - STRING

- DISCOUNT
  - DECIMAL
  - >= 0
  - DEFAULT 0

Chaves Candidatas={ID}

Entidade Moradas:

Descrição de moradas

- ADDRESS(ID[PK], STREET, ZIP_CODE, CITY,COUNTRY_CODE[FK]);
  - ID[PK]
    - INTEIRO
  - STREET
    - STRING
  - ZIP_CODE
    - STRING
  - CITY
    - STRING
  - COUNTRY_ID[FK]
    - INTEIRO

Chaves Candidatas={ID}
Chaves Estrangeiras={COUNTRY_ID}

Entidade País:

Descrição de Países

- COUNTRY(ID[PK], NAME);
  - ID[PK]
    - INTEIRO
  - NAME
    - STRING

Chaves Candidatas={ID}

Relação Pratos_Ingredientes:

Relaciona um prato com os ingredientes que compõem o mesmo

- COURSES_INGREDIENTS(COURSES_ID[FK], INGREDIENTS_ID[FK], QTY);
  - COURSES_ID[FK]
    - INTEIRO
  - INGREDIENTS_ID[FK]
    - INTEIRO
  - QTY
    - DECIMAL
    - DEFAULT 0

//VERIFICAR PREÇO QUANDO HÁ ALTERAÇÕES DE PREÇO NA COMPRA DE INGREDIENTES.

Chaves Candidatas={COURSES_ID}
Chaves Estrangeiras={COURSES_ID},{INGREDIENTS_ID}

Relação Pratos_Cardápio:

Relaciona um Cardápio com os pratos que compõem os mesmos

- MENU_COURSES(COURSES_ID[FK],MENU_ID[FK]);
  - COURSES_ID[FK]
    - INTEIRO
  - MENU_ID[FK]
    - INTEIRO

Chaves Candidatas={COURSES_ID}
Chaves Estrangeiras={COURSES_ID},{MENU_ID}

Relação Amigos:

Relacionas os amigos registados

- FRIENDS(REGISTERED_ID1[FK],REGISTERED_ID2[FK]) ;
  - REGISTERED_ID1[FK]
    - INTEIRO
  - REGISTERED_ID2[FK]
    - INTEIRO

Chaves Candidatas={REGISTERED_ID1,REGISTERED_ID2}
Chaves Estrangeiras={REGISTERED_ID1},{REGISTERED_ID2}

Relação Eventos_Amigos:

Relaciona os amigos que estão convidados para um evento

- EVENT_FRIENDS(BOOKING_ID[FK], CUSTOMER_ID[FK], STATUS);
  - BOOKING_ID[FK]
    - INTEIRO
  - CUSTOMER_ID[FK]
    - INTEIRO
  - STATUS
    - INTEIRO
    - {0,1} (0: not confirmed; 1: confirmed)
    - DEFAULT 0

Chaves Candidatas={BOOKING_ID,CUSTOMER_ID}
Chaves Estrangeiras={BOOKING},{CUSTOMER_ID}

Modelo de Dados Físico

Nesta secção procedemos à implementação em linguagem SQL dos scripts de criação e de eliminação do modelo relacional desenvolvido.

Dada a extensão dos scripts, os mesmos se encontram em anexo estes têm a seguinte denominação:

- *droptables_tp1.sql* – Script de eliminação de dados da base de dados
- *create_tables_tp1.sql* – Script de criação de estrutura de dados das tabelas da base de dados
- *data_feed_tp1.sql* – Script de população de dados na tabela
- *createdb-si2-tp1.sql* – Script que apaga a base de dados por forma a garantir que todos os *ids IDENTITY* da nossa base de dados iniciados a 1 como definido nas tabelas, create_tables_tp1.

Para garantir o correcto funcionamento da aplicação e uma maior simplicidade de execução de scripts, criamos um batch que executa todos os ficheiros sql definidos por nós neste trabalho. A baixo ilustramos um excerto deste:

```
@echo off
set server=localhost\SQLEXPRESS
set db=SI2_TP1
set dbmaster=master
ECHO ------------------------------------------------------------------------------ >>output.txt
ECHO Processing scripts... %TIME%  >output.txt
REM sqlcmd.exe  -S localhost\SQLEXPRESS -E   -d SI2_TP1 -i run_all.sql >>output.txt

ECHO ------------------------------------------------------------------------------ >>output.txt
ECHO Droping tables and data base SI2_TP1... >>output.txt
sqlcmd.exe  -S %server% -E -d %db% -i %CD%\1-droptables_tp1.sql >>output.txt
ECHO Done! >>output.txt

ECHO ------------------------------------------------------------------------------ >>output.txt
ECHO Creating data base SI2_TP1... >>output.txt
sqlcmd.exe  -S %server% -E -d %dbmaster% -i %CD%\2-createdb-si2-tp1.sql >> output.txt
ECHO Done! >>output.txt

ECHO ------------------------------------------------------------------------------ >>output.txt
ECHO Creating tables... >>output.txt
sqlcmd.exe  -S %server% -E -d %db% -i %CD%\3-create_tables_tp1.sql >>output.txt
ECHO Done! >>output.txt
```

Codificação em T-SQL

Tentámos codificar os requisitos da forma mais eficiente utilizando a linguagem T-SQL na implementação dos stored procedures, tricggers, etc.
Novamente, como a implementação de cada alínea é extensa, indicamos qual o ficheiro anexo que corresponde a implementação de cada alínea.

Criação de Cardápio

Ex4-a.sql

Criação de Evento Gastronómico

Ex4-b.sql

Sistema de Reservas

Ex4-c.sql

Valor do Evento

Ex4-d.sql

Cativação de Stock

Ex4-e.sql

Lista de Stock em

Ex4-f.sql

Actualização de Stock

Ex4-g.sql

Alarme de Validade

Ex4-h.sql

Listagem de Clientes

Ex4-i.sql

Listagem de Ingredientes Consumidos

Ex4-j.sql

Cardápios Lucrativos

Ex4-k.sql

Aplicação ADO.NET

"Implemente recorrendo à tecnologia ADO .NET uma aplicação que permita marcar reuniões com os clientes que atribuíram notas fora das expectativas. Use igualmente a tecnologia ADO .NET para exemplificar o uso dos procedimentos armazenados implementados as restantes alíneas. "
O ficheiro DB.cs tem a implementação do solicitado. A aplicação gráfica está longe de estar perfeita, mas é apenas uma simples interface de iteração.
O ficheiro tem os seguintes métodos:

- public IEnumerable<Comment> AllComments(), NegativeComments() e CommentsWithoutMeetng;
    - Lê dados relativos a comentários da base de dados e devolve um enumerável para ser tratado pela aplicação.
    - Cada método retorna respectivamente, todos os comentários, todos os comentários com avaliação negativa e comentários com reuniões por marcar.

- public Boolean insertMeeting(int comment_id, String date, String details)
    - Insere na base de dados os valores para uma nova reunião.
    - Actualizada o comentário com o status a 1 como forma de sinalizar que já foi marcada uma reunião.
    - Utilização do controlo transaccional para garantir que quando ocorre um erro na inserção ou na actualização, como por exemplo, chave estrangeira incorrecta, as alterações seja canceladas não deixando tuplos incoerentes na base de dados.

Para testar a execução dos procedimentos efecutamos estes metodos para teste.

- public static Boolean CreateCourses(String Name, int active, Double price)
    - Teste da alinea 4.a

Abaixo é ilustrado os comandos da mesma:

```
-Comands:
        +Show All Comments - 0
        +Show All Negative Comments - 1
        +Show Comments Without Meeting - 2
        +New Meeting - 3
        +New Course - 4
        +Exit - 5

Insert Command Please:
0
Comments:
        Serial Number: 1
        Customer:1
        Course: 1
        Grade: 1
        Comment: Sabor a detergentes!
-----------------------------------------
        Serial Number: 2
        Customer:1
        Course: 1
        Grade: 1
        Comment: Sabor a detergentes!
-----------------------------------------
        Serial Number: 3
        Customer:2
        Course: 1
        Grade: 1
        Comment: Sabor a detergentes!
-----------------------------------------

Insert Command Please:
-
```

Legenda 2 : Lista de todos os comentários.

```
Insert Command Please:
3
Comment Serial Number:
2
Day:
12
Month:
09
Year:
2011
Details:
Reuniao com Chefes de cozinha...
Insert Meeting


End. Press Enter.
```

Legenda 3 : Introducão de uma reunião.

```
Insert Command Please:
2
Comments:
        Serial Number: 3
        Customer:2
        Course: 1
        Grade: 1
        Comment: Sabor a detergentes!
-----------------------------------------
```

Legenda 4 : Comentários ao qual têm reuniões pendentes.

Como anteriormente foi criada uma reunião para o ID de comentário 2, então este já não está pendente.

Parte 2

Requesitos

Novo Modelo EA:

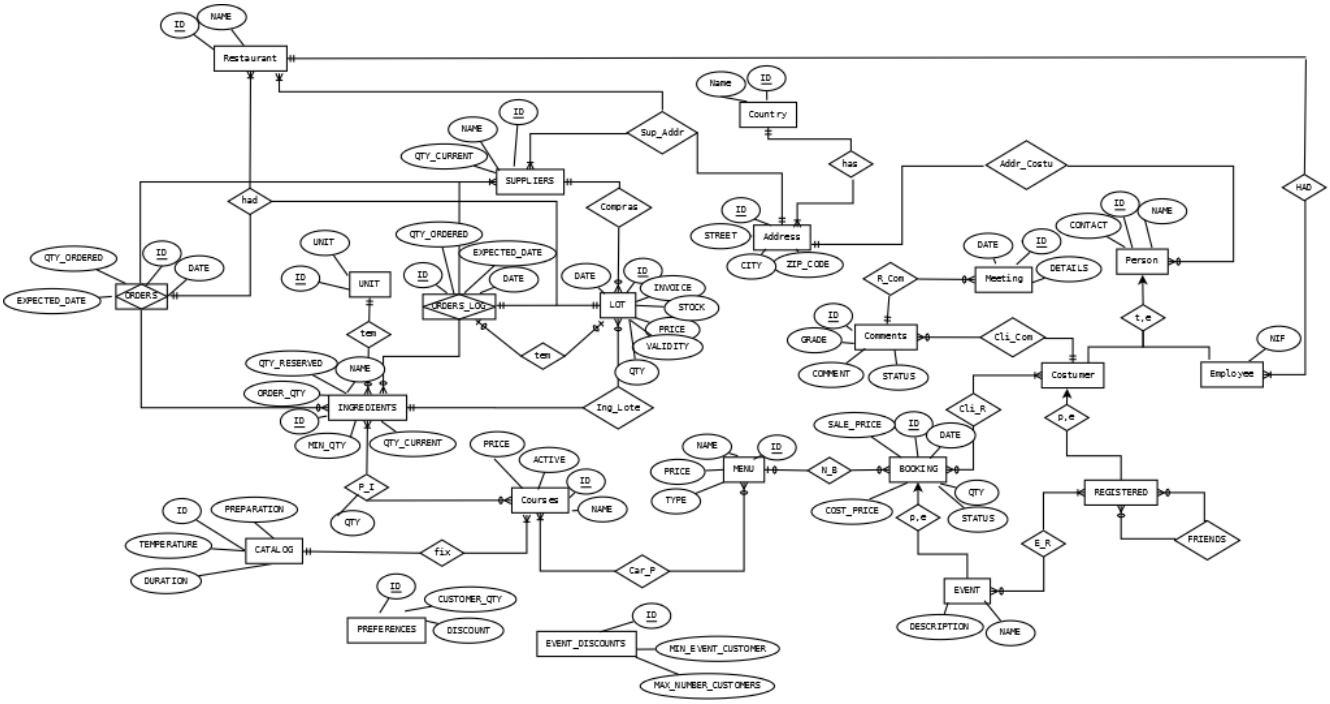

Figura 5: Esboço do Modelo Conceptual – Parte 2

Novas Entidades:

Entidade Restaurante:

Informação relativa a cada restaurante da cadeira de franchising.

- RESTAURANT(ID[PK], NAME, ADSRESS_ID[FK]);
  - ID
    - INTEIRO IDENTITY(1,1)
  - NAME
    - VARCHAR(50)
  - ADDRESS_ID
    - INTEIRO

Chaves Candidatas={ID}
Chaves Estrangeiras={ADDRESS_ID}

Entidade Person:

Informação relativa a cada pessoa, seja cliente ou funcionário.

- PERSON(ID[PK], NAME, CONTACT);
  - ID
    - INTEIRO IDENTITY(1,1)
  - NAME
    - CHAR(50)
  - CONTACT
    - VARCHAR(50)

Chaves Candidatas={ID}

Entidade Employee:

Informação relativa a cada funcionário.

- EMPLOYEE(PERSON_ID[PK], NIF, ADDRESS_ID[FK], RESTAURANT_ID[FK]);
  - PERSON_ID
    - INTEIRO IDENTITY(1,1)
  - NIF
    - INT
    - NOT NULL
  - ADDRESS_ID
    - INT
    - NOT NULL
  - RESTAURANT_ID
    - INT
    - NOT NULL

Chaves Candidatas={PERSON_ID}{NIF}
Chaves Estrangeiras={RESTAURANT_ID}{ADDRESS_ID}

Modificações em Entidades já existentes:

Entidade Lote:

Descrição para compras em Lotes

- LOT(ID[PK], INGREDIENTS_ID[FK], SUPPLIER_ID[FK], INVOICE, DATE, QTY, PRICE, VALIDITY, STOCK,RESTAURANT_ID[FK]);
  - ID[PK]
    - INTEIRO
  - INGREDIENTS_ID[FK]
    - INTEIRO
  - SUPPLIER_ID[FK]
    - INTEIRO
  - INVOICE
    - INTEIRO
  - DATE
    - DATA
  - QTY
    - DECIMAL
    - DEFAULT 0
  - PRICE
    - DECIMAL
    - DEFAULT 0
  - VALIDITY
    - DATA
  - STOCK
    - DECIMAL
    - DEFAULT 0
  - RESTAURANT_ID
    - INT
    - NOT NULL

Chaves Candidatas={ID}
Chaves Estrangeiras={INGREDIENTS_ID}{SUPPLIER_ID}{RESTAURANT_ID}

Entidade Encomendas:

Descrição de um encomenda

- ORDERS(ID[PK],SUPPLIER_ID[FK], INGREDIENT_ID[FK], DATE, QTY_ORDERED, EXPECTED_DATE, RESTAURANT_ID[FK],);
  - ID[PK]
    - INTEIRO
  - SUPPLIER_ID[FK]
    - INTEIRO
  - INGREDIENT_ID[FK]
    - INTEIRO
  - DATE
    - DATA
  - QTY_ORDERED
    - INTEIRO
  - EXPECTED_DATE
    - DATA
  - RESTAURANT_ID
    - INT
    - NOT NULL

Chaves Candidatas={ID}
Chaves Estrangeiras={SUPPLIER_ID}{INGREDIENT_ID}{RESTAURANT_ID}

Entidade Encomendas_Log:

Mantém o histórico de encomendas.

- ORDERS_LOG(ID[PK],SUPPLIER_ID[FK], INGREDIENT_ID[FK], DATE, QTY_ORDERED, EXPECTED_DATE, LOT_ID[FK], RESTAURANT_ID[FK],);
  - ID[PK]
    - INTEIRO
  - SUPPLIER_ID[FK]
    - INTEIRO
  - INGREDIENT_ID[FK]
    - INTEIRO
  - DATE
    - DATA
  - QTY_ORDERED
    - INTEIRO
  - EXPECTED_DATE
    - DATA
  - RESTAURANT_ID
    - INT
    - NOT NULL

Chaves Candidatas={ID}
Chaves Estrangeiras={SUPPLIER_ID}{INGREDIENT_ID}{RESTAURANT_ID}

Codificação XML

Alínea 1

Criação de documentos XML que exemplifiquem os casos de uso, por exemplo, a nota de transferência de stock, o curriculum do funcionário com as respectivas competências e a receita culinária.

- StockTransfer.xml
- Competence.xml
- Catalog.xml

Alínea 2

Criação de um DTD que formalize as estruturas de dados do ponto anterior.

- StockTransfer.dtd
- Catalog.dtd
- Competence.dtd

Alínea 3

Criação de XSDs que refinem os DTDs anteriores identificando os aspectos que foram melhorados.

- StockTransfer.xsd
- Catalog.xsd
- Competence.xsd

O XML Schema Definition permite-nos estruturar o ficheiro com as restrições já definidas na alínea anterior bem como adicionar-lhe:

- Tipo de valor:
  - Integer;
  - String;
  - Decimal;
  - Date

Alínea 4

Geração de um Script SQL, a partir da nota de transferência de stock, para proceder à respectiva actualização na base de dados central.

- UpdateStockTransfer.sql

Alínea 5

Geração, em conformidade com o XSD e a partir da base de dados central, de um documento XML com a lista de competências e quantidade dos funcionários de um determinado restaurante.

competences.xsd

Alínea 6

Implementação de mecanismos de detecção de fraude que identifiquem irregularidades ou aspectos relevantes para o negócio protagonizadas pelos fornecedores, funcionários ou clientes.

Conclusão

No decorrer deste trabalho, pensamos que alcançamos os objetivos propostos. O trabalho é interessante e levamos a tomar em atenção algumas situações na implementação .

O controlo transacional revelou-se uma peça fulcral na devida execução de procedimentos armazenados, como seria de esperar.

O que se tornou revigoraste no inicio com a implementação de uma nova tecnologia, tornou-se depressa uma frustração, pois dado ao nível de inexperiência com a mesma, uma vez que o objetivo obtido não era o esperado tendo que mudar de "lógica" processual algumas vezes para que o mesmo funcionasse.

De forma geral a experiência foi interessante.

Bibliografia

Davidson, Louis, Pro SQL Server 2008 Relational Database Design and Implementation

Filipe, Porfírio, Slides de apoio à unidade curricular, Triggers, AdoDotNet, Transacções, XML, DTD, XPath, XSLT, XSD, XQuery e DOM XML API.

SQL Server Books Online, Março 2011, http://msdn.microsoft.com/en-us/library/ms130214.aspx

Transaction-SQL Refence, Março 2011, http://msdn.microsoft.com/en-us/library/ms189826%28v=SQL.90%29.aspx