



TRABAJO DOMICILIARIO: BASES DE DATOS APLICADA I			
FECHA:	27/10/2024		
ALUMNO/A:	Dylan Pointis, Facundo Mercado y Lucas Figueredo		
LEGAJO:	-	DNI:	-
CURSO:	2024-090-3-A-M	TURNO:	Mañana
CARRERA:	Ingeniería en Sistemas Informáticos		
PROFESOR/A:	Martín M. Rivas Sarquis		
MODALIDAD:	Domiciliario		

**UNIDADES POR EVALUAR DEL PROGRAMA DE LA MATERIA:**

- Unidad 2: Modelando un Data Warehouse.
- Unidad 3: Introducción a minería de datos.

**CRITERIOS DE RESOLUCIÓN**

Los alumnos/as recibirán la consigna del examen en la fecha de evaluación prevista por el cronograma.

El examen constará de 2 instancias

1. Entrega de las consignas y explicación de la metodología de evaluación por parte del docente a los alumnos/as.

Duración del examen: 2 (dos) semanas: Los/as alumnos/as deberán entregar la evaluación digital por parte de los alumnos/as al docente.

2. Defensa coloquial oral grupal.


**INTRODUCCIÓN (100 PTS)**

Debe obtener al menos 60/100 para la aprobación de esta parte.

Para llevar adelante esta evaluación domiciliaria grupal, se pensó la siguiente dinámica a desarrollar de 2 (dos) semanas de trabajo. El cual los alumnos tendrán que investigar y llevar adelante la consigna. La evaluación finalizara con una exposición Grupal Oral en defensa y explicación del Trabajo desarrollado.

## SAKILA DATABASE

<https://github.com/uai-argentina/bda-2024-q2/tree/main/sakila-db>

 Fue construido originalmente para el motor MySQL, por lo que los nombres de las tablas y los campos están todas en minúscula y cada separación de palabra es dividida por un guion bajo. Esta convención no es la que recomienda Microsoft.

Considérese que su **grupo de trabajo** elegido representa una consultora de sistemas especialista en el desarrollo y construcción de **Data Warehouse** y **Minería de datos**. La cual es contratada para analizar el negocio de **Sakila** que viene funcionando hace más de 10 años. Este simula un sistema de gestión de alquiler de películas y proporciona un entorno que representa las operaciones diarias de una tienda de alquiler de películas.

Los alumnos deberán utilizar como usuario clave del negocio (**Key User**) a [ChatGPT](#), el cual le responderá las consultas funcionales y técnicas del modelo de datos de Sakila. Con el fin de entender y comprender el negocio que se está relevando.

El proyecto se llevará adelante en Fases/Etapas y se debe ir documentando cada una de ellas hasta llegar a la construcción de un **Data Warehouse**.

El resultado final junto con la documentación y todo el proyecto debe ser subido a un **Github** provisto por la universidad o el alumno. De carácter público, que contendrá los siguientes artefactos.

- Documentación del proyecto.
- Solución de Visual Studio.
- Script de creación de Esquemas y objetos necesarios para el Data Warehouse.
- Información sobre el grupo de trabajo, materia, recomendaciones y conclusión.

## Lineamientos Generales

1. Descarga los Script de SQL de creación del esquema de datos y el de inserción de datos de ejemplo.
2. A partir de la base de datos Sakila (OLTP) de datos transaccionales, será necesario relevar el modelo e identificar y construir un Data Warehouse del mismo. Para esto se utilizará la solución de Integration Services (SSIS) para la construcción de ETL.
  - a. Utilizar las convenciones recomendadas por Microsoft.  
<https://github.com/microsoft/sql-server-samples/tree/master/samples/da>

[tabases](#)

- b. Utilizar Visual Studio con la extensión de **Integration Services** para el desarrollo de ETL.
  - c. Utilizar una base de datos de **Staging Area** para la construcción de los ETL.
3. Construir los **Data Marts** que crean convenientes.
4. **Documente** todos los pasos y fases de la construcción del Data Warehouse.
5. Elegir un Data Mart y construir un CUBO multidimensional con la solución de **Analysis Services**.
6. Utilizar **Power BI Desktop** para conectar al Cubo construido en el punto anterior. El mismo se podrá utilizar desde la herramienta.

## Links Útiles

- **Analysis Service**

- **Visual Studio 2019**

- <https://marketplace.visualstudio.com/items?itemName=ProBITools.MicrosoftAnalysisServicesModelingProjects>

- **Visual Studio 2022**

- <https://marketplace.visualstudio.com/items?itemName=ProBITools.MicrosoftAnalysisServicesModelingProjects2022>

- **Integration Service**

- **Visual Studio 2019**

- <https://marketplace.visualstudio.com/items?itemName=SSIS.SqlServerIntegrationServicesProjects>

- **Visual Studio 2022**

- <https://marketplace.visualstudio.com/items?itemName=SSIS.MicrosoftDataToolsIntegrationServices>

- **SQL Server Data Tools**

- <https://learn.microsoft.com/en-us/sql/ssdt/download-sql-server-data-tools-ssdt?view=sql-server-ver15>

- **SQL Server Developer Edition**

- [SQL Server 2022 Developer Edition](#)
  - [SQL Server 2019 Developer Edition](#)

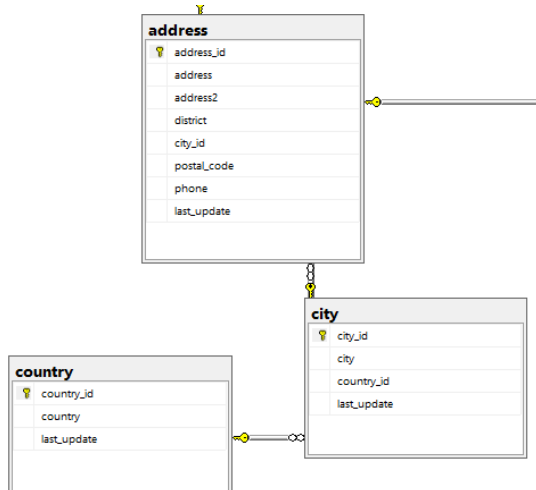
- **Power BI Desktop**

- <https://www.microsoft.com/en-us/download/details.aspx?id=58494>

## Etapas I: Análisis de los Sistemas de Información

En esta primera etapa, se analizó la fuente de datos para comprender su integración y evaluar su calidad. Es importante encontrar datos incompletos o errores y comprenderlos para garantizar que el Data Warehouse se construya sobre una base sólida de datos confiables y bien integrados.

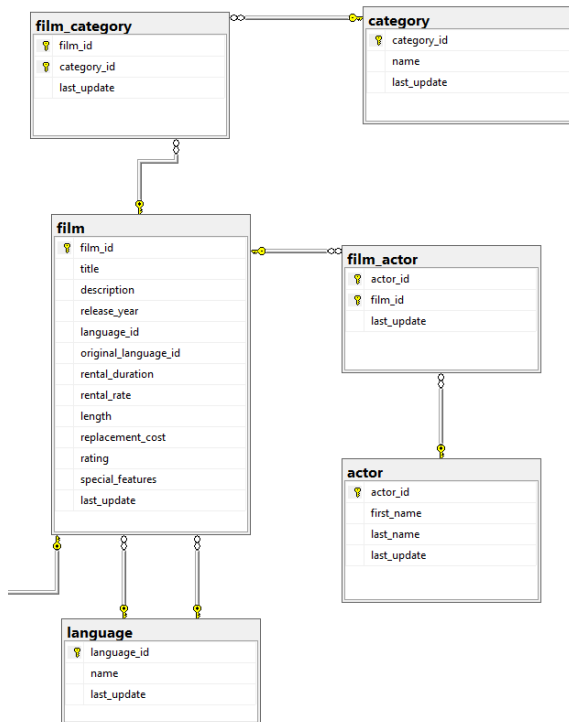
Durante el análisis de los datos originales pudimos observar que la tabla “address” presenta una clave foránea con ciudad, que a su vez se conecta con país



Asimismo, notamos que las columnas “address2”, “district” y “phone” no son utilizadas, por lo que son datos incompletos que no pueden ser integrados correctamente en el Data Warehouse

Results		Messages						
	address_id	address	address2	district	city_id	postal_code	phone	last_update
1	1	47 MySakila Drive	NULL		300	NULL		2006-02-15 04:45:30.000
2	2	28 MySQL Boulevard	NULL		576	NULL		2006-02-15 04:45:30.000
3	3	23 Workhaven Lane	NULL		300	NULL		2006-02-15 04:45:30.000
4	4	1411 Lilydale Drive	NULL		576	NULL		2006-02-15 04:45:30.000
5	5	1913 Hanoi Way	NULL		463	35200		2006-02-15 04:45:30.000
6	6	1121 Loja Avenue	NULL		449	17886		2006-02-15 04:45:30.000
7	7	692 Joliet Street	NULL		38	83579		2006-02-15 04:45:30.000
8	8	1566 Ined Manor	NULL		349	53561		2006-02-15 04:45:30.000

Con respecto a la tabla Film, se analizó cómo se relaciona con otras tablas. Puede tener un lenguaje y un lenguaje original, puede tener una o varias categorías y una o varios actores



Al consultar sus registros, identificamos que el campo “original\_language\_id” no es utilizado en ningún de ellos, por lo que también se considera como un dato incompleto.

	film_id	title	description	release_year	language_id	original_language_id	rental_duration	rental_rate	length	replacement_cost	rating	special_features
1	1	ACADEMY DINOSAUR	A Epic Drama of a Feminist And a Mad Scientist who ...	2006	1	NULL	6	0.99	86	20.99	PG	Deleted Scenes, Behind the Scenes
2	2	ACE GOLDFINGER	A Astounding Epistle of a Database Administrator And ...	2006	1	NULL	3	4.99	48	12.99	G	Trailers, Deleted Scenes
3	3	ADAPTATION HOLES	A Astounding Reflection of a Lumberjack And a Car w...	2006	1	NULL	7	2.99	50	18.99	NC-17	Trailers, Deleted Scenes
4	4	AFFAIR PREJUDICE	A Fanciful Documentary of a Frisbee And a Lumberjac...	2006	1	NULL	5	2.99	117	26.99	G	Commentaries, Behind the Scenes
5	5	AFRICAN EGG	A Fast-Paced Documentary of a Pastry Chef And a De...	2006	1	NULL	6	2.99	130	22.99	G	Deleted Scenes
6	6	AGENT TRUMAN	A Intrepid Panorama of a Robot And a Boy who must ...	2006	1	NULL	3	2.99	169	17.99	PG	Deleted Scenes
7	7	AIRPLANE SIERRA	A Touching Saga of a Hunter And a Butler who must ...	2006	1	NULL	6	4.99	62	28.99	PG-13	Trailers, Deleted Scenes
8	8	AIRPORT POLLOCK	A Epic Tale of a Moose And a Girl who must Confront ...	2006	1	NULL	6	4.99	54	15.99	R	Trailers
9	9	ALABAMA DEVIL	A Thoughtful Panorama of a Database Administrator A...	2006	1	NULL	3	2.99	114	21.99	PG-13	Trailers, Deleted Scenes
10	10	ALADDIN CALENDAR	A Action-Packed Tale of a Man And a Lumberjack wh...	2006	1	NULL	6	4.99	63	24.99	NC-17	Trailers, Deleted Scenes
11	11	AI AMO VIDEOTAPF	A Boring Fable of a Butler And a Cat who must Fight a	2006	1	NULL	6	0.99	126	16.99	G	Commentaries, Behind the Scenes

## Etapla II: Especificación de Requerimientos

Se contacta con el usuario clave del negocio para entender los requisitos necesarios y relevantes para la organización Sakila. Se identifican los hechos, los datos más importantes para el negocio y la toma de decisiones. Filtrando lo que es realmente útil para los usuarios finales.

En el negocio Sakila las rentas son los datos más importantes, serán los hechos de interés primario a analizar y para la toma de decisiones.

Con respecto a la tabla “film”, al estudiarla podemos identificar atributos no relevantes para el negocio como “description”, “replacement\_cost”, “special\_features”, “last\_update” o su relación con actores. Por lo que, al no ser considerados útiles para el negocio, se decidió omitirlos.

En el caso de la tabla “customer”, se decidió no utilizar los campos de “store\_id”, “active”, “create\_date”, “last\_update”. Consideramos que no aportan valor para el análisis o toma de decisiones.

Para la tabla Address se excluyó el campo “last\_update”.

Para la tabla Staff se decidió no incluir “username”, “active”, “password”, “last\_update”.

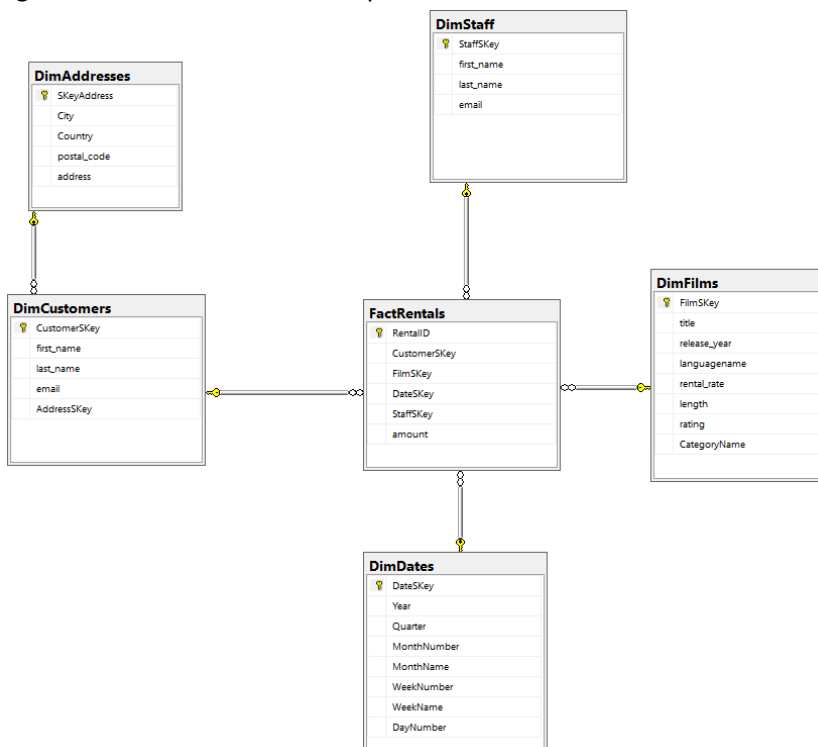
### **Etapas III: Diseño Conceptual**

Se empieza a construir el esqueleto del Data WareHouse, eligiendo la tabla central de hechos “FactRental” e incluyendo su medida “amount”.

Se decidió utilizar un esquema copo de nieve, o en inglés snowflake, para obtener un nivel parcial de normalización, una estructura más organizada y evitar cierta duplicación de datos. Se crean jerarquías de dimensiones, conectando una dimensión con otra.

Se crean las dimensiones de “DimCustomers”, “DimFilms”, “DimDates”, “DimAddress” y “DimStaff” para detallar los hechos del negocio.

Se optó por obtener el nombre del lenguaje e incluirlo directamente en la Dimension Films, en lugar de hacer una dimensión aparte.



## Etapa IV: Refinamiento y Validación

Se refinó el esquema diseñado en la etapa anterior, para obtener un Data WareHouse que funcione de manera más eficiente y se adapte mejor a las necesidades de la organización.

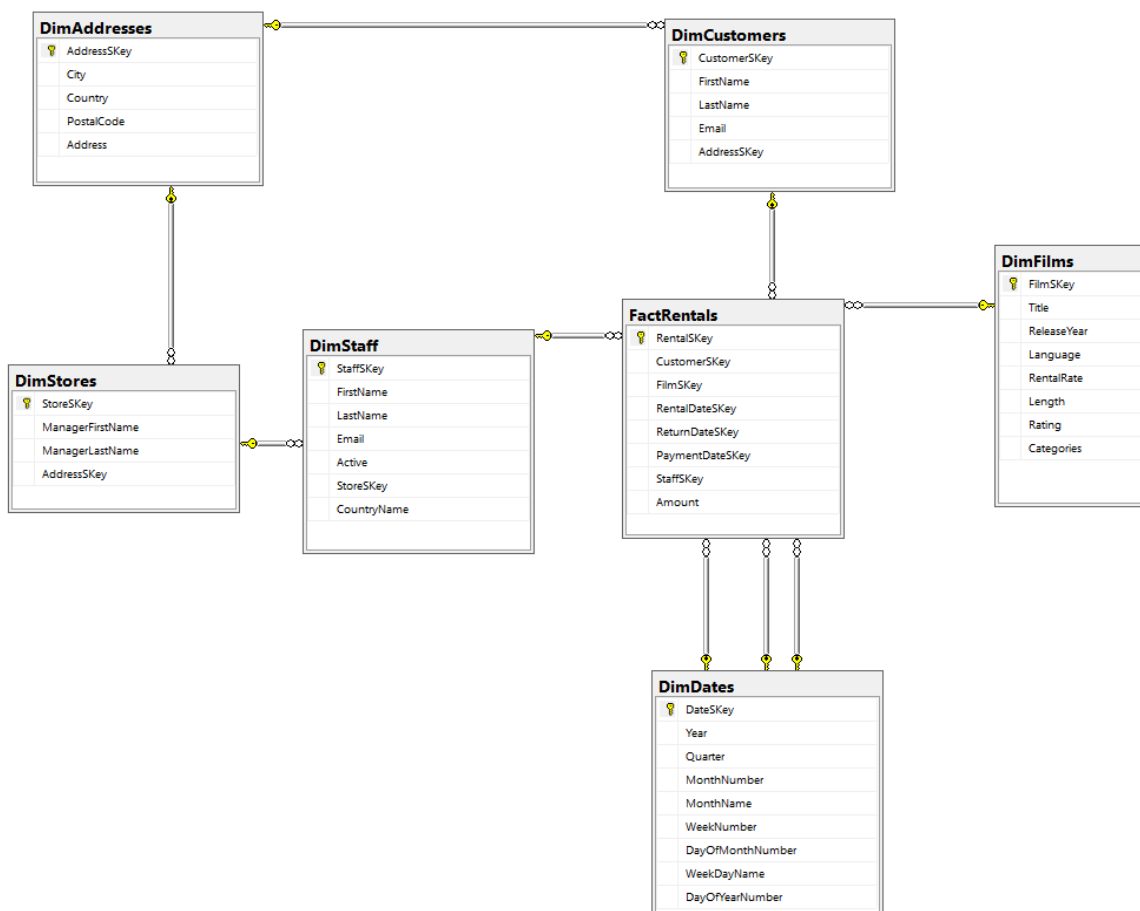
Tras realizar las primeras cargas de trabajo preliminares, se optó por representar las categorías de una película concatenarlas en un campo "Categories".

También se implementaron las convenciones y nomenclaturas recomendadas por Microsoft, así como se validaron los tipos datos para que puedan cargarse correctamente los datos.

## Etapa V: Diseño Lógico

Se añade la dimensión Store, considerando que es fundamental a la hora de hacer análisis útiles para el negocio. La dimensión "Staff" se conectó con "Store", de esta forma, podemos acceder a los datos de la tienda desde el empleado que participó en la renta. Así como también, obtener información sobre los empleados de cada tienda.

Además, se optó por agregar un atributo "CountryName" a la dimensión Staff, pues representa el único dato geográfico útil para los empleados de la tienda.



## Etapa VI: Dise

no Fisico

Se implement

o el diseno conceptual llevado a cabo en las etapas anteriores en un entorno fisico en SQL Server, utilizando el modelo Snowflak, configurando las tablas y sus relaciones en base al diseno definido, consiguiendo cierto nivel de normalizacion.

Asimismo, se configuraron todos los tipos de datos para que puedan ser completamente compatibles entre la base de datos transaccional y el Data Warehouse construido.

Tras completar la implementaci

on fisica, se realiza el proceso ETL para poblar las tablas del Data Warehouse con datos historicos, utiles para la toma de decisiones y consultas analticas en cubos multidimensionales.

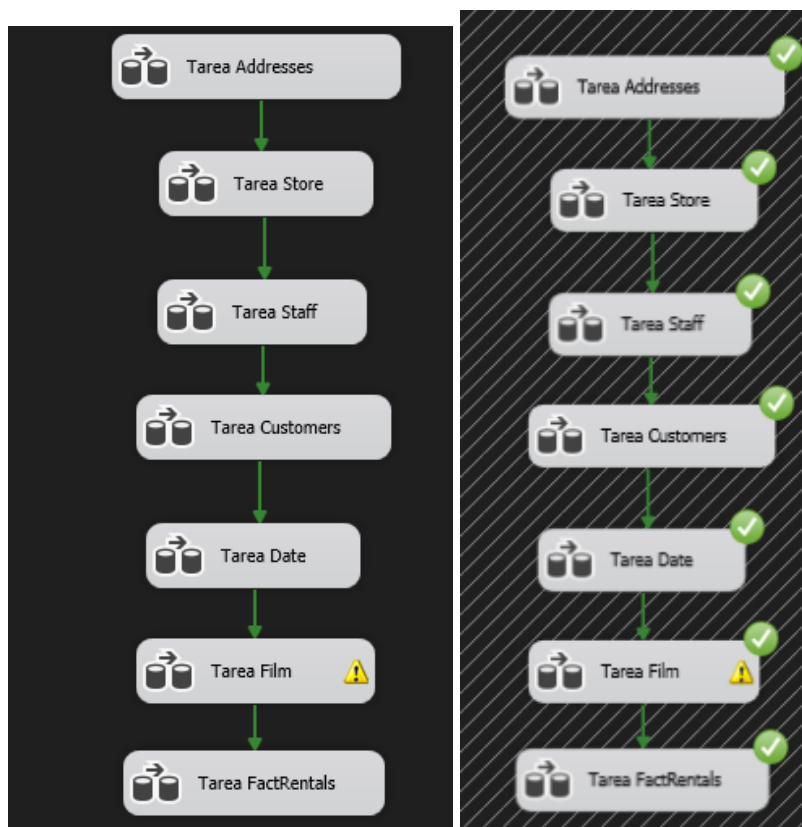
## PROCESO ETL

Se describe el proceso ETL (Extract, Transform, Load) para cargar los datos desde la base de datos "sakila" al Data WareHouse "DWSakila", realizado en Visual Studio con la extensi

on de Integration Services.

Antes de cargar los datos en la tabla Fact, primero debemos cargar las dimensiones.

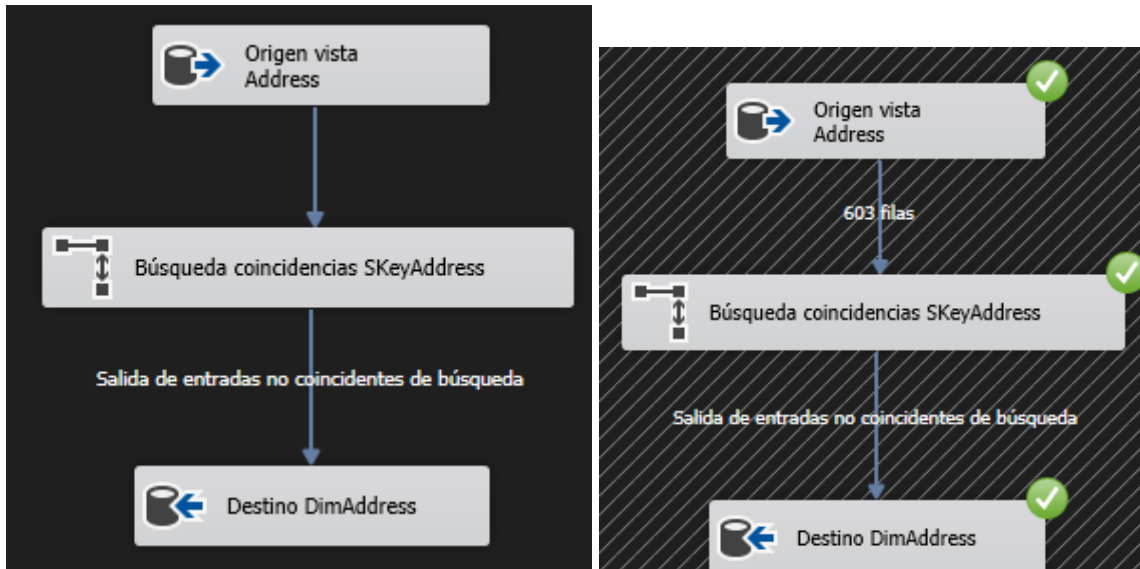
El flujo es el siguiente:





## DimAddresses

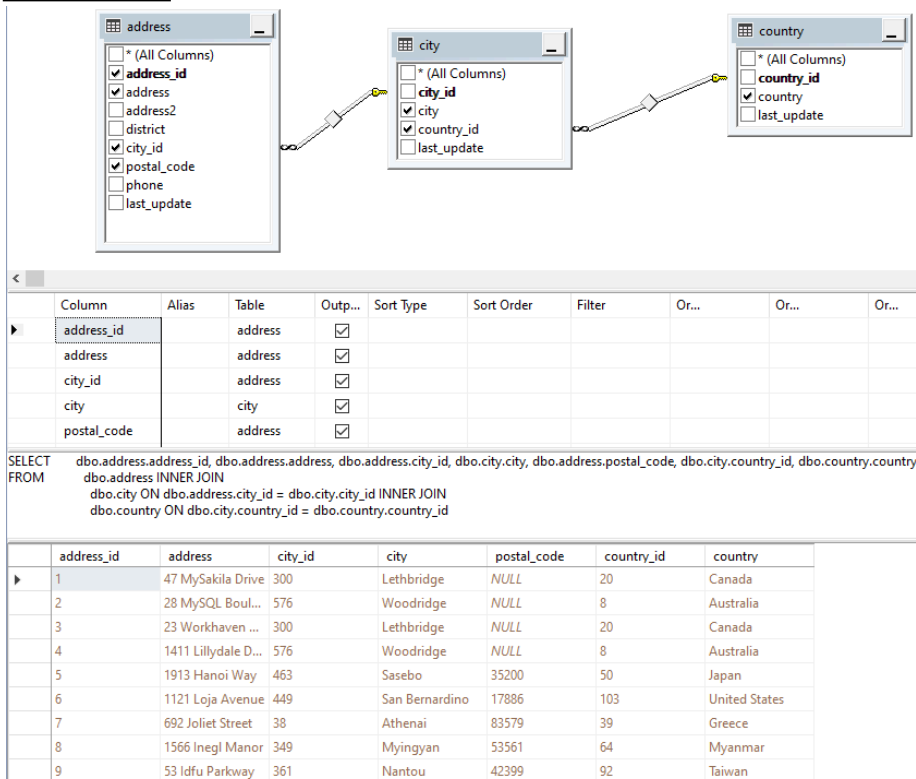
Primero cargamos la dimensión DimAddresses con sus datos



Se implementó una vista para facilitar el acceso a los datos deseados. También se implementó un elemento Búsqueda o “Lookup” para evitar cargar nuevamente registros que ya existan en la dimensión Address del Data WareHouse, evitando conflictos de clave primaria.

Para cargar el resto de las dimensiones, también se utilizaron vistas y el elemento LookUp, con el mismo propósito

### Vista Address:



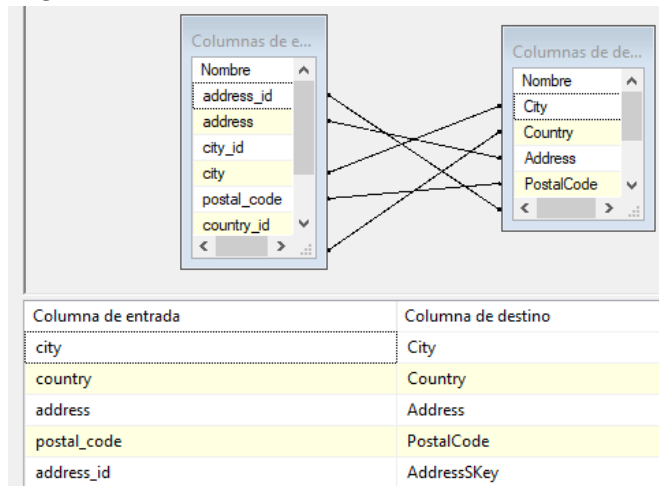
Column	Alias	Table	Outp...	Sort Type	Sort Order	Filter	Or...	Or...	Or...
address_id		address	<input checked="" type="checkbox"/>						
address		address	<input checked="" type="checkbox"/>						
city_id		address	<input checked="" type="checkbox"/>						
city		city	<input checked="" type="checkbox"/>						
postal_code		address	<input checked="" type="checkbox"/>						

```

SELECT
FROM
    dbo.address.address_id, dbo.address.address, dbo.address.city_id, dbo.city.city, dbo.address.postal_code, dbo.city.country_id, dbo.country.country
    dbo.address INNER JOIN
    dbo.city ON dbo.address.city_id = dbo.city.city_id INNER JOIN
    dbo.country ON dbo.city.country_id = dbo.country.country_id
  
```

	address_id	address	city_id	city	postal_code	country_id	country
1	47	MySakila Drive	300	Lethbridge	NULL	20	Canada
2	28	MySQL Boul...	576	Woodridge	NULL	8	Australia
3	23	Workhaven ...	300	Lethbridge	NULL	20	Canada
4	1411	Lillydale D...	576	Woodridge	NULL	8	Australia
5	1913	Hanoi Way	463	Sasebo	35200	50	Japan
6	1121	Loja Avenue	449	San Bernardino	17886	103	United States
7	692	Joliet Street	38	Athenai	83579	39	Greece
8	1566	Inegl Manor	349	Myingyan	53561	64	Myanmar
9	53	Idfu Parkway	361	Nantou	42399	92	Taiwan

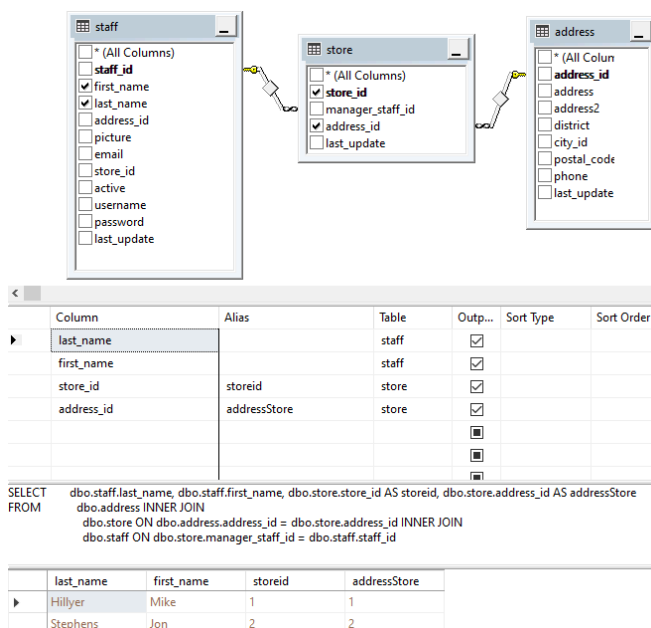
## Asignaciones:



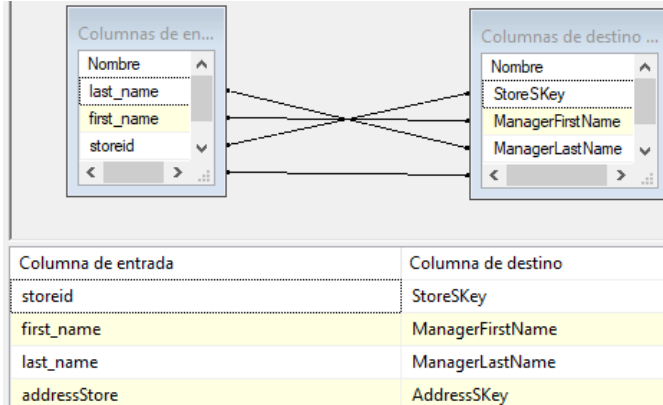
## DimStore



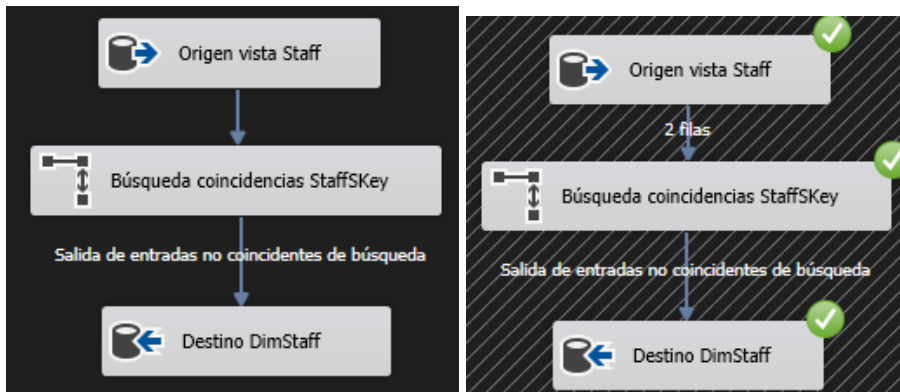
## Vista Store:



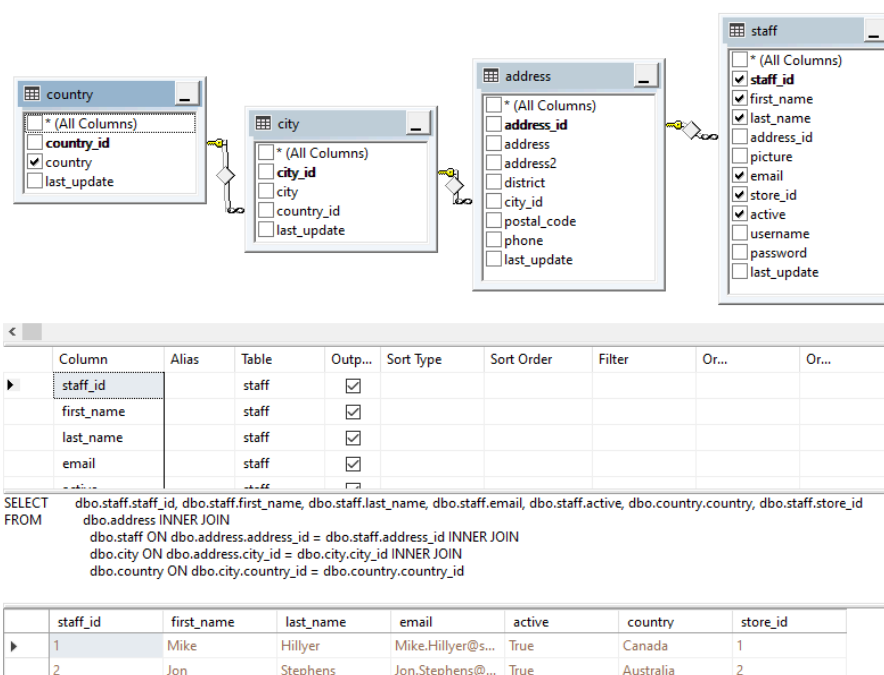
## Asignaciones:



## DimStaff



## Vista Staff:



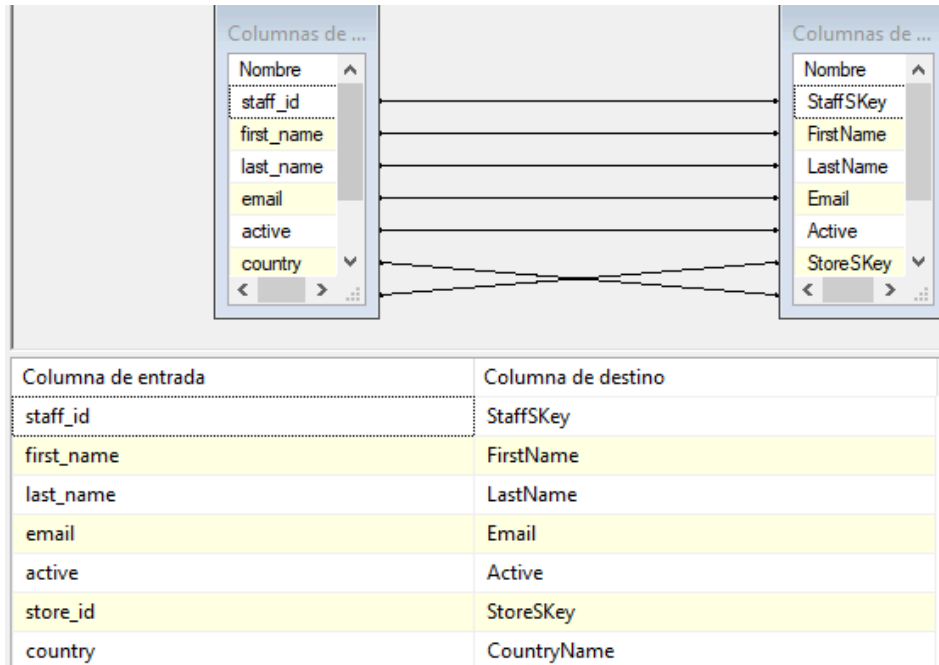


# UAI

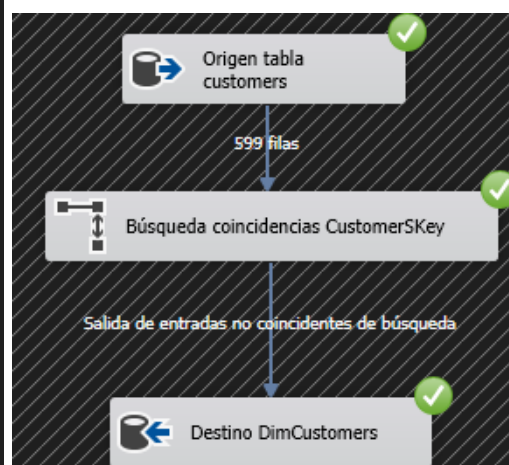
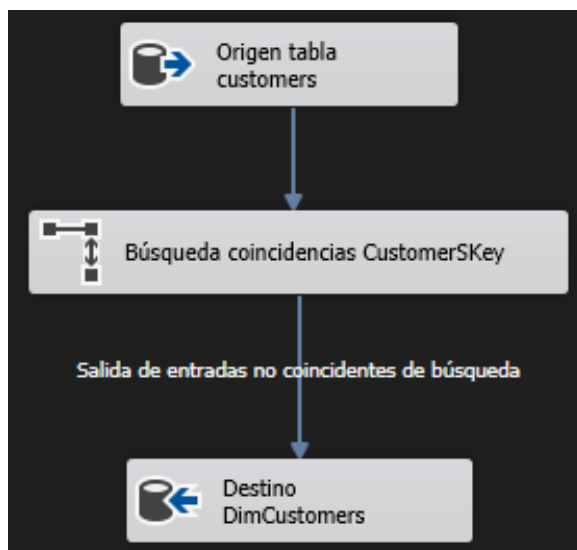
Universidad Abierta  
Interamericana



Asignaciones:



## DimCustomers



En este caso, no fue necesario utilizar una vista. Directamente utilizamos la tabla para cargar los datos.

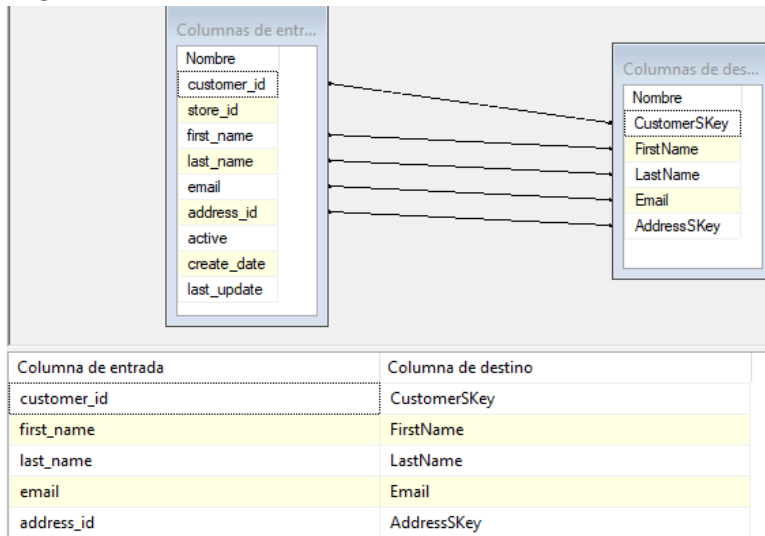


# UAI

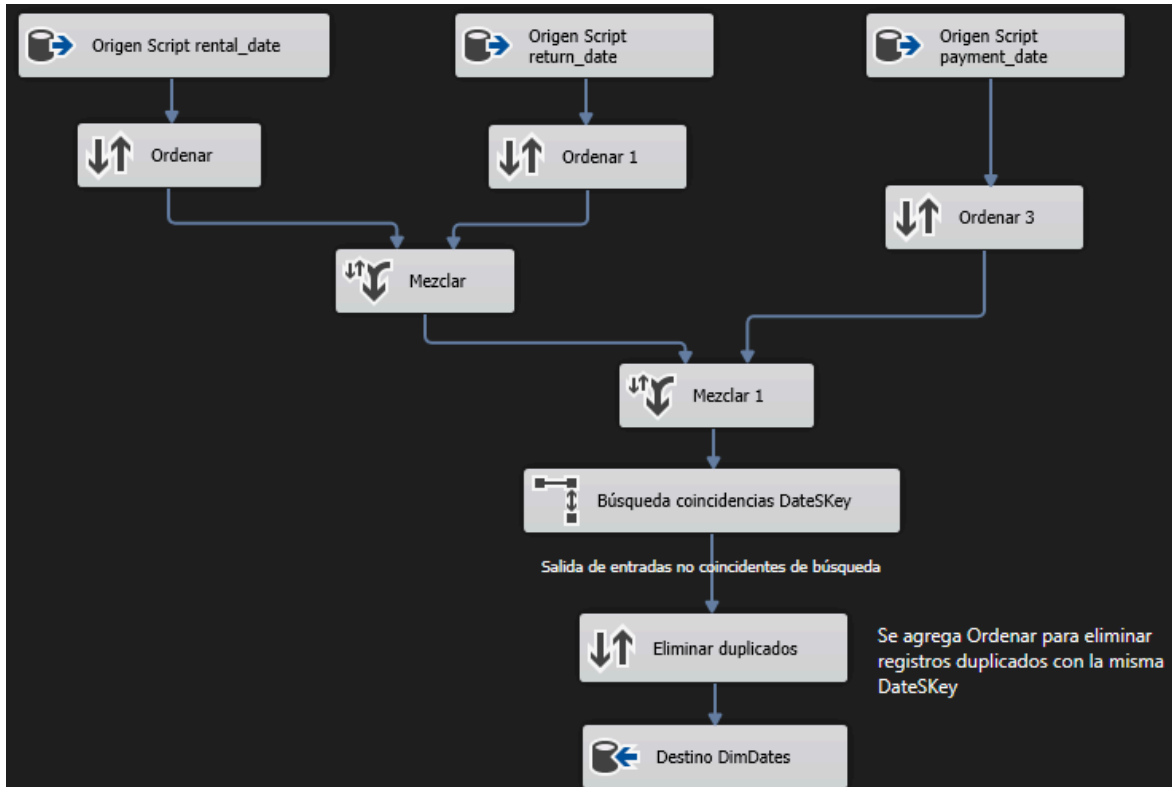
Universidad Abierta  
Interamericana



## Asignaciones:



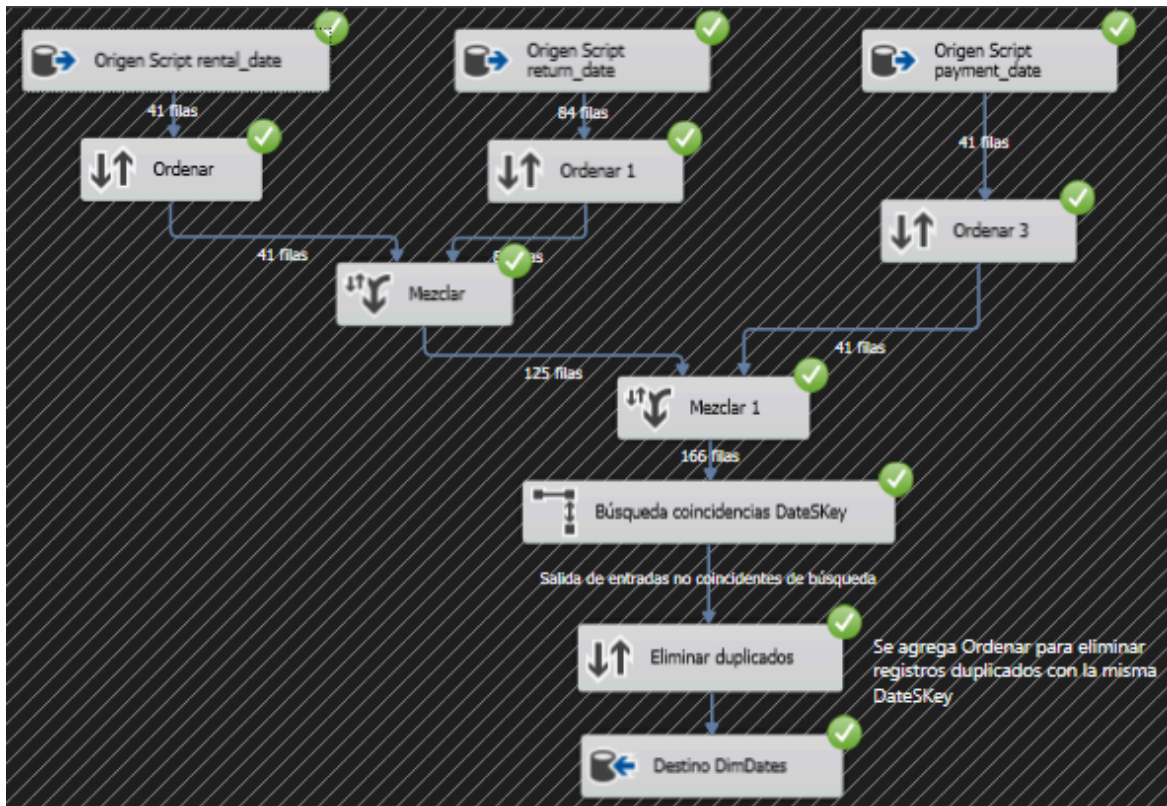
## DimDates





# UAI

Universidad Abierta  
Interamericana



Se utilizó un script para convertir las fechas “rental\_date” de tipo DateTime de la base de datos original al tipo de dato INT y así utilizarlas como clave primaria. También para extraer el Año, Cuatrimestre, Número de Mes, Nombre de mes, Número de semana, Nombre del día, Número del día, Número del día del año

```
SELECT
    DISTINCT CONVERT(INT, FORMAT(Origin.rental_date, 'yyyyMMdd')) AS 'DateSKey',
    YEAR(Origin.rental_date) AS 'Year',
    DATEPART(Quarter, Origin.rental_date) AS 'Quarter',
    MONTH(Origin.rental_date) AS MonthNumber,
    DATENAME(MONTH, Origin.rental_date) AS 'MonthName',
    DATEPART(WEEK, Origin.rental_date) AS WeekNumber,
    DATENAME(weekday, Origin.rental_date) AS WeekDayName,
    DATEPART(DAY, Origin.rental_date) AS DayOfMonthNumber,
    DATEPART(dayofyear, Origin.rental_date) AS DayOfYearNumber
FROM [sakila].[dbo].[rental] AS Origin
LEFT JOIN [DWSakila].[dbo].[DimDates] AS Dest
ON CAST(Origin.rental_date AS INT) = Dest.DateSKey
WHERE Dest.DateSKey IS NULL
```

	DateSKey	Year	Quarter	MonthNumber	MonthName	WeekNumber	WeekDayName	DayOfMonthNumber	DayOfYearNumber
1	20060214	2006	1	2	Febrero	8	Martes	14	45
2	20050731	2005	3	7	Julio	31	Domingo	31	212
3	20050728	2005	3	7	Julio	31	Jueves	28	209
4	20050816	2005	3	8	Agosto	34	Martes	16	228
5	20050802	2005	3	8	Agosto	32	Martes	2	214
6	20050730	2005	3	7	Julio	31	Martes	30	207

Se repitió el mismo script para la fecha de retorno “return\_date” y para la fecha de pago

“payment\_date”.

Se utilizó el elemento mezclar para combinar todos los resultados en un mismo set de datos. El cual luego fue ordenado para eliminar fechas duplicadas.

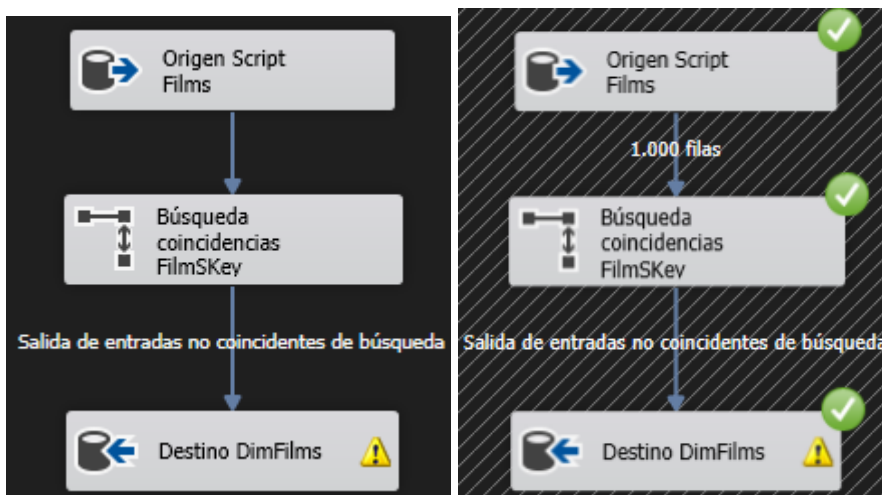
Asignaciones:

Columnas de ent...	Columnas de des...
Nombre	Nombre
DateSKey	DateSKey
Year	Year
Quarter	Quarter
MonthNumber	MonthNumber
MonthName	MonthName
WeekNumber	WeekNumber
WeekDayN...	DayOfMont...
< >	< >

Columna de entrada	Columna de destino
DateSKey	DateSKey
Year	Year
Quarter	Quarter
MonthNumber	MonthNumber
MonthName	MonthName
WeekNumber	WeekNumber
DayOfMonthNumber	DayOfMonthNumber
WeekDayName	WeekDayName
DayOfYearNumber	DayOfYearNumber

## DimFilm



Se utilizó un script para conseguir los detalles de la película, consiguiendo su lenguaje y concatenando sus categorías en el campo “Categories”



## Script

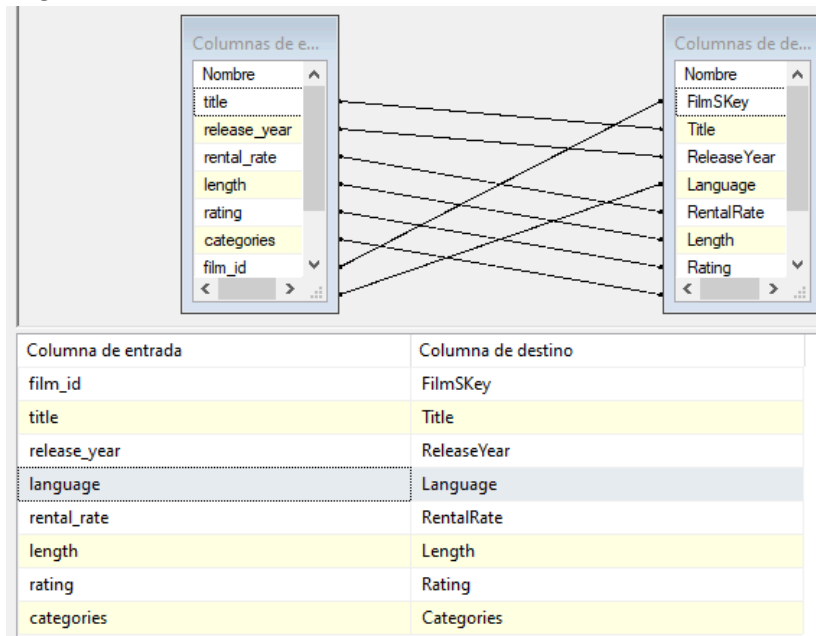
```
SELECT
  f.film_id,
  f.title,
  f.release_year,
  l.name AS language,
  f.rental_rate,
  f.length,
  f.rating,
  STRING_AGG(c.name, ', ') AS categories
FROM film f
INNER JOIN language l
  on f.language_id = l.language_id
INNER JOIN film_category fc ON f.film_id = fc.film_id
INNER JOIN category c ON fc.category_id = c.category_id
GROUP BY f.film_id, f.title, f.release_year, l.name, f.rental_rate, f.length, f.rating, c.name
```

115 %

Results Messages

	film_id	title	release_year	language	rental_rate	length	rating	categories
1	1	ACADEMY DINOSAUR	2006	English	0.99	86	PG	Documentary
2	2	ACE GOLDFINGER	2006	English	4.99	48	G	Horror
3	3	ADAPTATION HOLES	2006	English	2.99	50	NC-17	Documentary
4	4	AFFAIR PREJUDICE	2006	English	2.99	117	G	Horror
5	5	AFRICAN EGG	2006	English	2.99	130	G	Family
6	6	AGENT TRUMAN	2006	English	2.99	100	PG	Family

## Asignaciones:





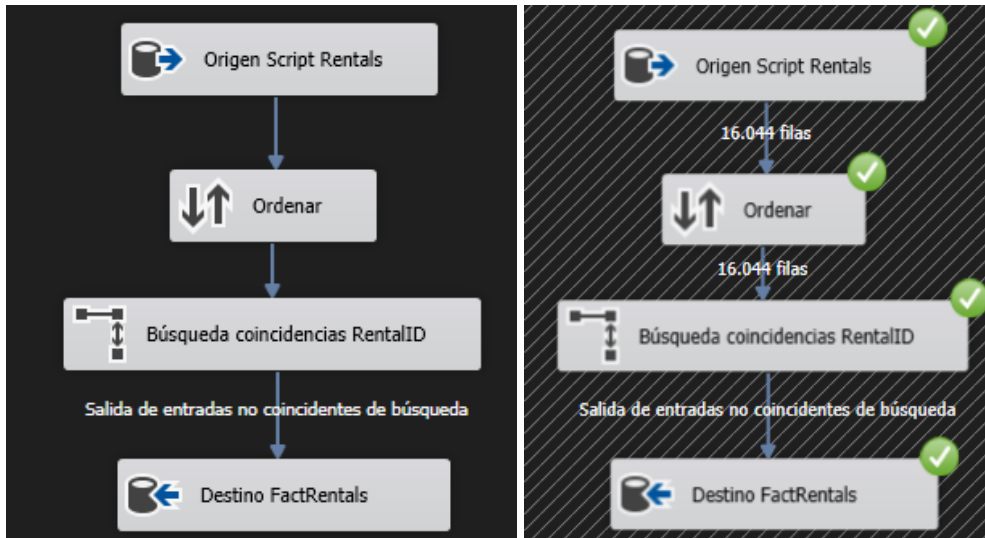


# UAI

Universidad Abierta  
Interamericana



## FactRentals



Se utilizó un script para convertir la fecha de renta, de retorno y de pago en INT y así cargarlos en RentalDateSKey, ReturnDateSKey y PaymentDateSKey. Además, se usó para unir con Inventory y conseguir el Film key de la renta

## Script

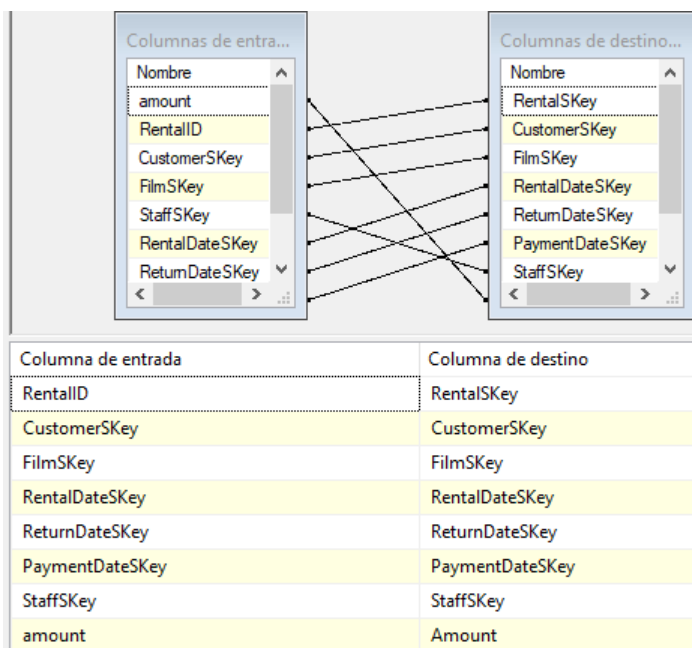
```
SELECT
dbo.rental.rental_id AS RentalID,
dbo.rental.customer_id AS CustomerSKey,
dbo.inventory.film_id AS FilmSKey,
dbo.rental.staff_id AS StaffSKey,
CONVERT(INT, FORMAT(dbo.rental.rental_date, 'yyyyMMdd')) AS RentalDateSKey,
CONVERT(INT, FORMAT(dbo.rental.return_date, 'yyyyMMdd')) AS ReturnDateSKey,
CONVERT(INT, FORMAT(dbo.payment.payment_date, 'yyyyMMdd')) AS PaymentDateSKey,
dbo.payment.amount
FROM dbo.payment INNER JOIN
dbo.rental ON dbo.payment.rental_id = dbo.rental.rental_id INNER JOIN
dbo.inventory ON dbo.rental.inventory_id = dbo.inventory.inventory_id
order by RentalID
```

115 %

Results Messages

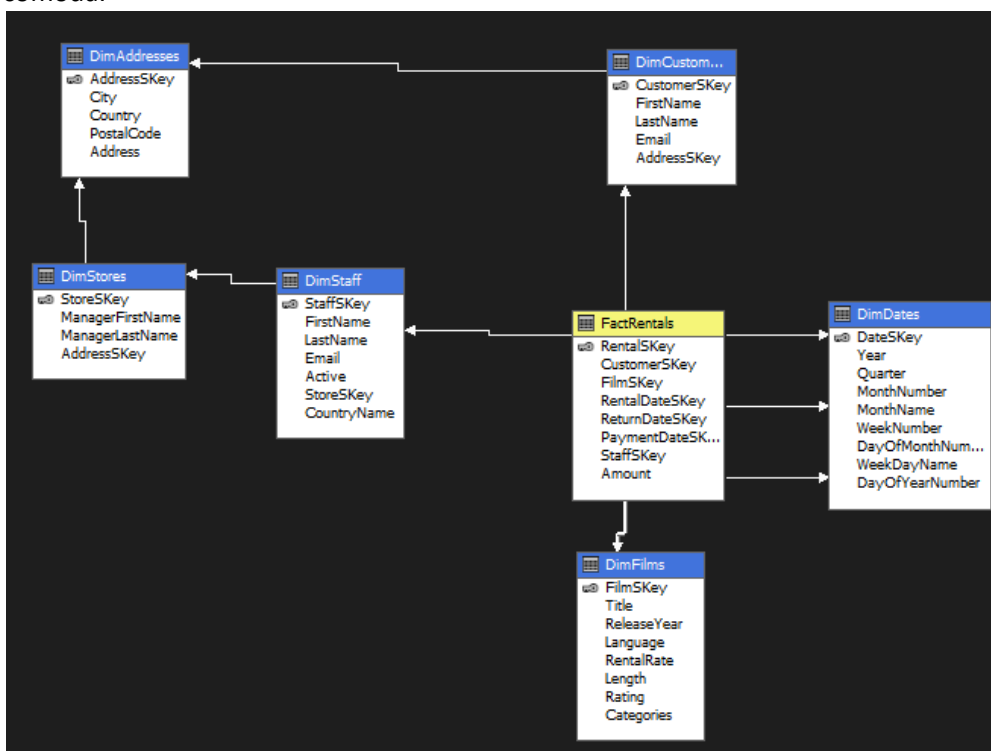
	RentalID	CustomerSKey	FilmSKey	StaffSKey	RentalDateSKey	ReturnDateSKey	PaymentDateSKey	amount
1	1	130	80	1	20050524	20050526	20050524	2.99
2	2	459	333	1	20050524	20050528	20050524	2.99
3	3	408	373	1	20050524	20050601	20050524	3.99
4	4	333	535	2	20050524	20050603	20050524	4.99
5	5	222	450	1	20050524	20050602	20050524	6.99

## Asignaciones



## CONSTRUCCION DE CUBO MULTIDIMENSIONAL

Utilizando la herramienta de Analysis Services se llevó a cabo la construcción del cubo multidimensional, para consultar los datos desde diferentes perspectivas, de una manera cómoda.



Estructura de cubo    Uso de dimensiones    Cálculos    KPI    Acciones    Particiones    Agregaciones    Perspectivas    Traducciones    Explorador

---

Editar como texto    Importar...    MDX    [Iconos]

### Metadatos

Buscar modelo

<<All>

- CuboDWSakila
  - Measures
    - Fact Rentals
      - Amount
      - Recuento Fact Rentals
  - KPI
  - Dim Customers
    - Address
    - Address S Key
    - City
    - Country
    - Customer S Key
    - Email
    - First Name
    - Last Name
    - Postal Code
  - Dim Films
  - Dim Staff
  - Payment Date S

Dimensión	Jerarquia	Operador	Expresión de filtro
<Seleccionar dimensión>			
Country	Amount		
Finland	101,74		
France	374,04		
French Guiana	103,78		
French Polynesia	235,46		
Gambia	129,7		
Germany	831,04		
Greece	232,46		
Greenland	137,66		
Holy See (Vatican City State)	152,66		
Hong Kong	142,7		
Hungary	111,71		
India	6628,28		
Indonesia	1510,33		
Iran	950,75		
Iraq	111,73		
Israel	422,01		
Italy	831,11		
Japan	3470,75		
Kazakhstan	198,48		
Kenya	253,46		
Kuwait	111,74		

En SQL Management Studio:

CuboDWSakila [Browse] SQLQuery7.sql - (Luu217\Navegador (56))\* 090L6PC22-102179.sakila - Diagram\_0\*

Edit as Text Import... MDX

**CuboDWSakila**

- Metadata
  - Search Model
  - Measure Group: <All>
  - CuboDWSakila
    - Measures
      - Fact Rentals
        - Amount
        - Recuento Fact Rentals
    - KPIs
      - Dim Customers
      - Dim Films
      - Dim Staff
        - Active
        - Email
        - First Name
        - Last Name
        - Manager First Name
        - Manager Last Name
        - Staff Country Name
        - Store S Key
        - Store S Key
        - StoreAddress
        - StoreAddress S Key
        - StoreCity
        - StoreCountry
        - StorePostal Code
      - Payment Date S
      - Rental Date S
      - Return Date S

Dimension	Hierarchy	Operator	Filter Expression
<Select dimension>			

Store S Key	Amount
1	33524,620000005
2	33881,9400000049

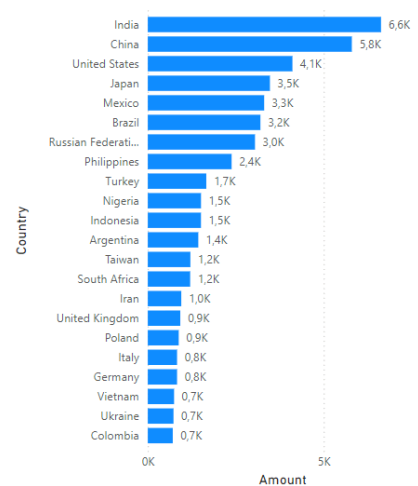
## POWER BI

Ejemplos de gráficos posibles a implementar:

Países de clientes que mas ingresos y cantidades de rentas generaron

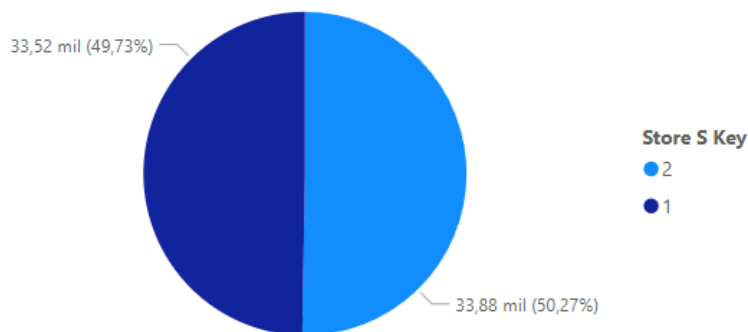
Country	Amount	Recuento Fact Rentals
India	6.628,28	1572
China	5.798,74	1426
United States	4.110,32	968
Japan	3.470,75	825
Mexico	3.307,04	796
Brazil	3.200,52	748
Russian Federation	3.045,87	713
Philippines	2.381,32	568
Turkey	1.662,12	388
Nigeria	1.511,48	352
Indonesia	1.510,33	367
Argentina	1.434,48	352
Taiwan	1.209,95	305
South Africa	1.204,15	285
Iran	950,75	225
United Kingdom	922,81	219
Poland	877,97	203
Italy	831,11	189
Germany	831,04	196
Vietnam	746,28	172
Ukraine	730,42	158
Colombia	709,41	159
Egypt	694,39	161
Venezuela	683,30	170
<b>Total</b>	<b>67.406,56</b>	<b>16044</b>

Amount by Country



Recaudación por tienda

Amount por Store S Key



Recaudación por tienda, año y trimestre

Store S Key	Amount	Year	Quarter
1	7.172,79	2005	2
1	26.133,66	2005	3
1	218,17	2006	1
2	7.280,54	2005	2
2	26.305,39	2005	3
2	296,01	2006	1
<b>Total</b>	<b>67.406,56</b>		