DATA WAREHOUSE IMPLEMENTATION PROJECT

Github Repo: https://github.com/divyagarigipati/datawarehouse/tree/main

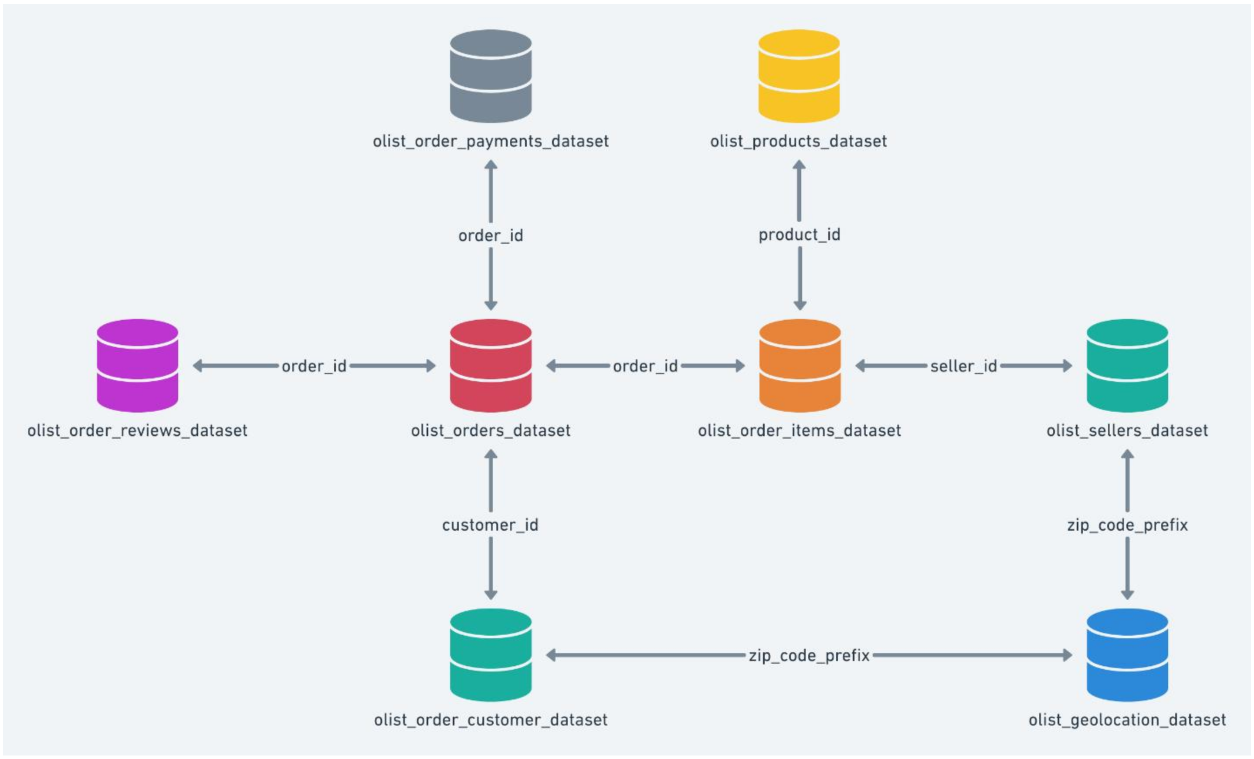
Notebook: https://databricks-prod-cloudfront.cloud.databricks.com/public/4027ec902e239c93eaaa8714f173bcfc/3969169828089009/2859625473491059/2340490761257204/latest.html

Data Source:

Raw data files

https://www.kaggle.com/datasets/olistbr/brazilian-ecommerce?select=olist_customers_dataset.csv

Initial Data Schema:



The raw e-commerce database has a relational structure, where the primary table is olist_orders_dataset which is an order lifecycle table combining different aspects. There is a olist_order_customer_dataset table where customer information is stored, and it is linked by customer_id. The olist_order_items_dataset includes order details and references olist_products_dataset` for the product details, as well as olist_sellers_dataset for seller information. In olist_geolocation_dataset, zip_code_prefix connects sellers to geographic data. Orders are linked to their payments in olist_order_payments_dataset and their reviews in olist_order_reviews_dataset. This schema provides a structured way to analyze the customer transactions, sellers' performance, delivery efficiency to fetch complete business insights.

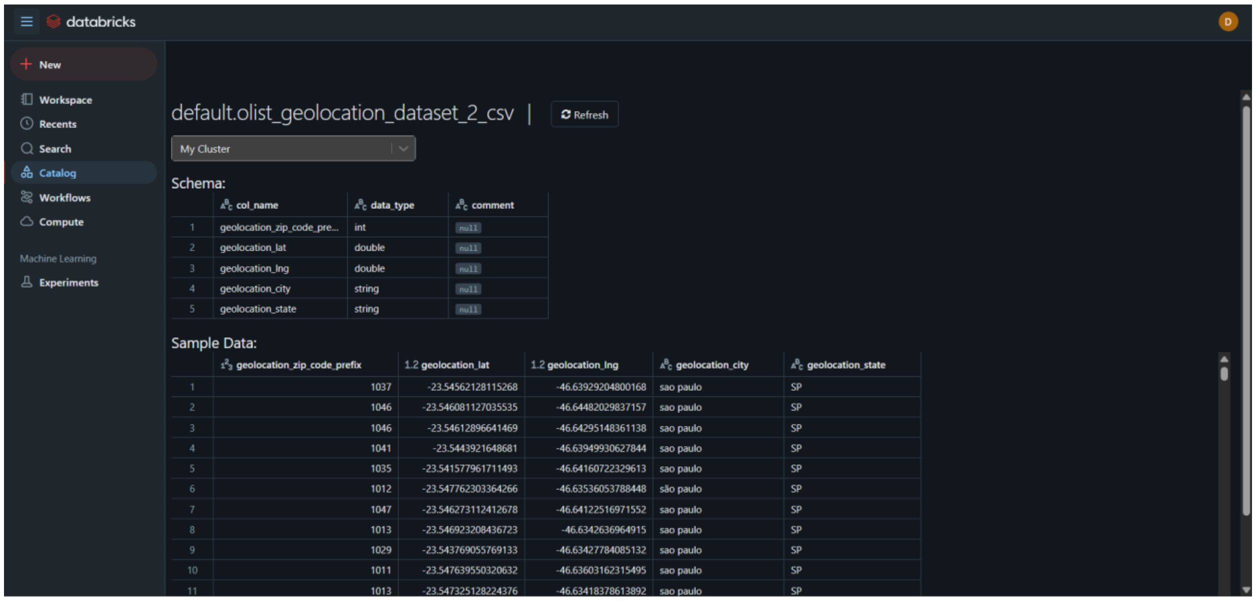
DATA WAREHOUSE IMPLEMENTATION PROJECT

Extracting Data:

Loading tables into data bricks hive metastore. The first row will be header, and schema will be inferred by databricks

The following images also contain the data dictionary (inferredschema) for each table.

Gelocations:



default.olist_geolocation_dataset_2_csv

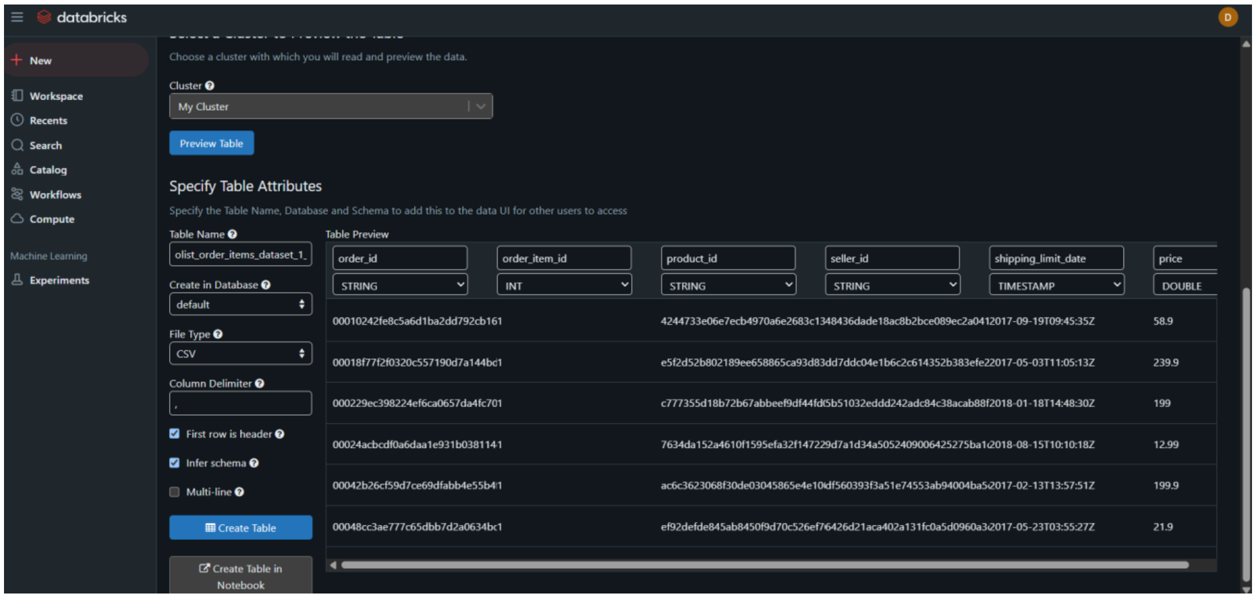
My Cluster

Schema:

	col_name	data_type	comment
1	geolocation_zip_code_pre...	int	null
2	geolocation_lat	double	null
3	geolocation_lng	double	null
4	geolocation_city	string	null
5	geolocation_state	string	null

Sample Data:

	geolocation_zip_code_prefix	geolocation_lat	geolocation_lng	geolocation_city	geolocation_state
1	1037	-23.54562128115268	-46.63929204800168	sao paulo	SP
2	1046	-23.546081127035535	-46.64482029837157	sao paulo	SP
3	1046	-23.54612896641469	-46.64295148361138	sao paulo	SP
4	1041	-23.5443921648681	-46.63949930627844	sao paulo	SP
5	1035	-23.541577961711493	-46.64160722329613	sao paulo	SP
6	1012	-23.547762303364266	-46.63536053788448	são paulo	SP
7	1047	-23.546273112412678	-46.64122516971552	sao paulo	SP
8	1013	-23.546923208436723	-46.6342636964915	sao paulo	SP
9	1029	-23.543769055769133	-46.63427784085132	sao paulo	SP
10	1011	-23.547639550320632	-46.63603162315495	sao paulo	SP
11	1013	-23.547325128224376	-46.63418378613892	sao paulo	SP



Choose a cluster with which you will read and preview the data.

Cluster: My Cluster

Specify Table Attributes

Specify the Table Name, Database and Schema to add this to the data UI for other users to access

Table Name: olist_order_items_dataset_1_

Create in Database: default

File Type: CSV

Column Delimiter: ,

First row is header: ✓

Infer schema: ✓

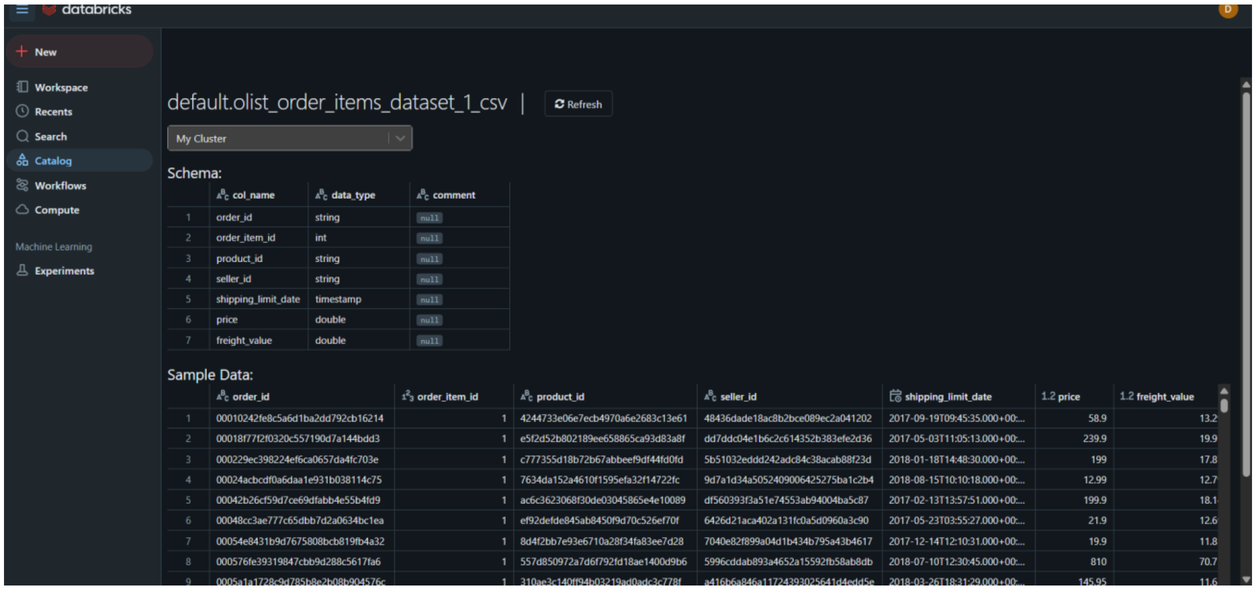
Multi-line: ☐

Table Preview

order_id	order_item_id	product_id	seller_id	shipping_limit_date	price
STRING	INT	STRING	STRING	TIMESTAMP	DOUBLE
00010242fe8c5a6d1ba2dd792cb161		4244733e06e7ecb4970a6e2683c1348436dade18ac8b2bce089ec2a0412017-09-19T09:45:35Z			58.9
00018f77f2f0320c557190d7a144bd1		e5f2d52b802189ee658865ca93d83dd7ddc04e1b6c2c614352b383efe22017-05-03T11:05:13Z			239.9
000229ec398224ef6ca0657da4fc701		c777355d18b72b67abbeef9df44fd5b51032eddd242adc84c38acab88f2018-01-18T14:48:30Z			199
00024acbcdf0a6daa1e931b0381141		7634da152a4610f1595efa32f147229d7a1d34a5052409006425275ba12018-08-15T10:10:18Z			12.99
00042b26cf59d7ce69dfabb4e55b41		ac6c3623068f30de03045865e4e10df560393f3a51e74553ab94004ba52017-02-13T13:57:51Z			199.9
00048cc3ae777c65dbb7d2a0634bc1		ef92defde845ab8450f9d70c526ef76426d21aca402a131fc0a5d0960a32017-05-23T03:55:27Z			21.9

DATA WAREHOUSE IMPLEMENTATION PROJECT

Order items:



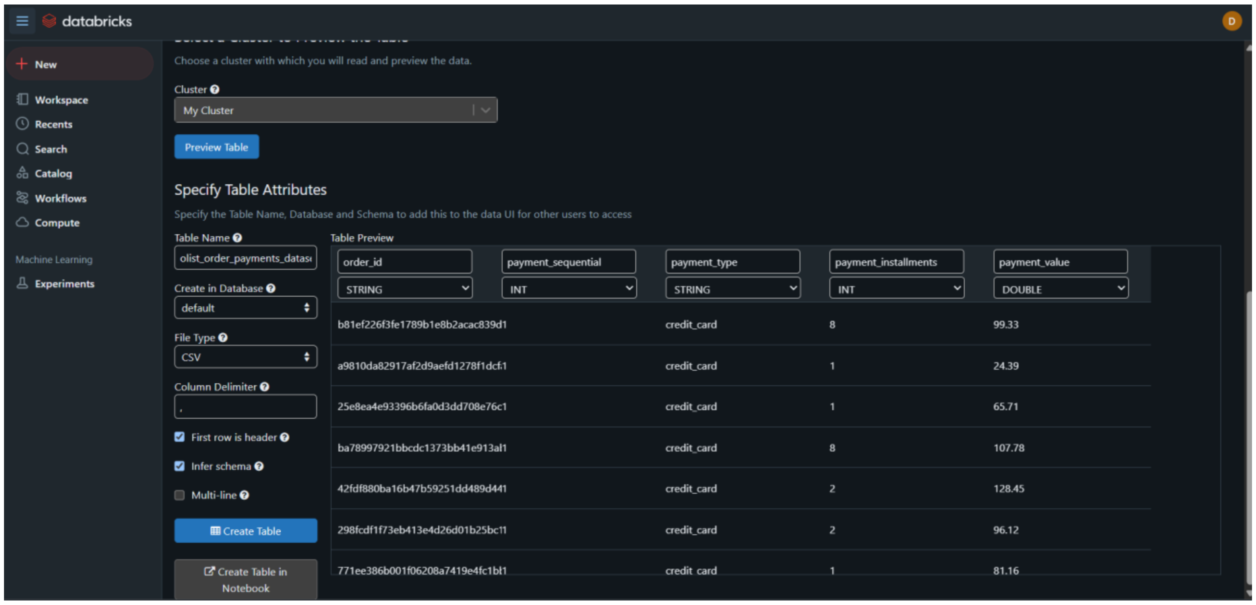
default.olist_order_items_dataset_1_csv

Schema:

	col_name	data_type	comment
1	order_id	string	null
2	order_item_id	int	null
3	product_id	string	null
4	seller_id	string	null
5	shipping_limit_date	timestamp	null
6	price	double	null
7	freight_value	double	null

Sample Data:

	order_id	order_item_id	product_id	seller_id	shipping_limit_date	price	freight_value
1	00010242fe8c5a6d1ba2dd792cb16214	1	4244733e06e7ecb4970a6e2683c13e61	48436dade18ac8b2bce089ec2a041202	2017-09-19T09:45:35.000+00:...	58.9	13.2
2	00018f77f2f0320c557190d7a144bdd3	1	e5f2d52b802189ee658865ca93d83a8f	dd7ddc04e1b6c2c614352b383efe2d36	2017-05-03T11:05:13.000+00:...	239.9	19.9
3	000229ec398224ef6ca0657da4fc703e	1	c777355d18b72b67abbeef9df44fd0fd	5b51032eddd242adc84c38acab88f23d	2018-01-18T14:48:30.000+00:...	199	17.8
4	00024acbcdf0a6daa1e931b038114c75	1	7634da152a4610f1595efa32f14722fc	9d7a1d34a5052409006425275ba1c2b4	2018-08-15T10:10:18.000+00:...	12.99	12.7
5	00042b26cf59d7ce69dfabb4e55b4fd9	1	ac6c3623068f30de03045865e4e10089	df560393f3a51e74553ab94004ba5c87	2017-02-13T13:57:51.000+00:...	199.9	18.1
6	00048cc3ae777c65dbb7d2a0634bc1ea	1	ef92defde845ab8450f9d70c526ef70f	6426d21aca402a131fc0a5d0960a3c90	2017-05-23T03:55:27.000+00:...	21.9	12.6
7	00054e8431b9d7675808bcb819fb4a32	1	8d4f2bb7e93e6710a28f34fa83ee7d28	7040e82f899a04d1b434b795a43b4617	2017-12-14T12:10:31.000+00:...	19.9	11.8
8	000576fe39319847cbb9d288c5617fa6	1	557d850972a7d6f792fd18ae1400d9b6	5996cddab893a4652a15592fb58ab8db	2018-07-10T12:30:45.000+00:...	810	70.7
9	0005a1a1728c9d785b8e2b08b904576c	1	310ae3c140ff94b03219ad0adc3c778f	a416b6a846a11724393025641d4edd5e	2018-03-26T18:31:29.000+00:...	145.95	11.6



Choose a cluster with which you will read and preview the data.

Cluster: My Cluster

Specify Table Attributes

Specify the Table Name, Database and Schema to add this to the data UI for other users to access

Table Name: olist_order_payments_datas

Create in Database: default

File Type: CSV

Column Delimiter: ,

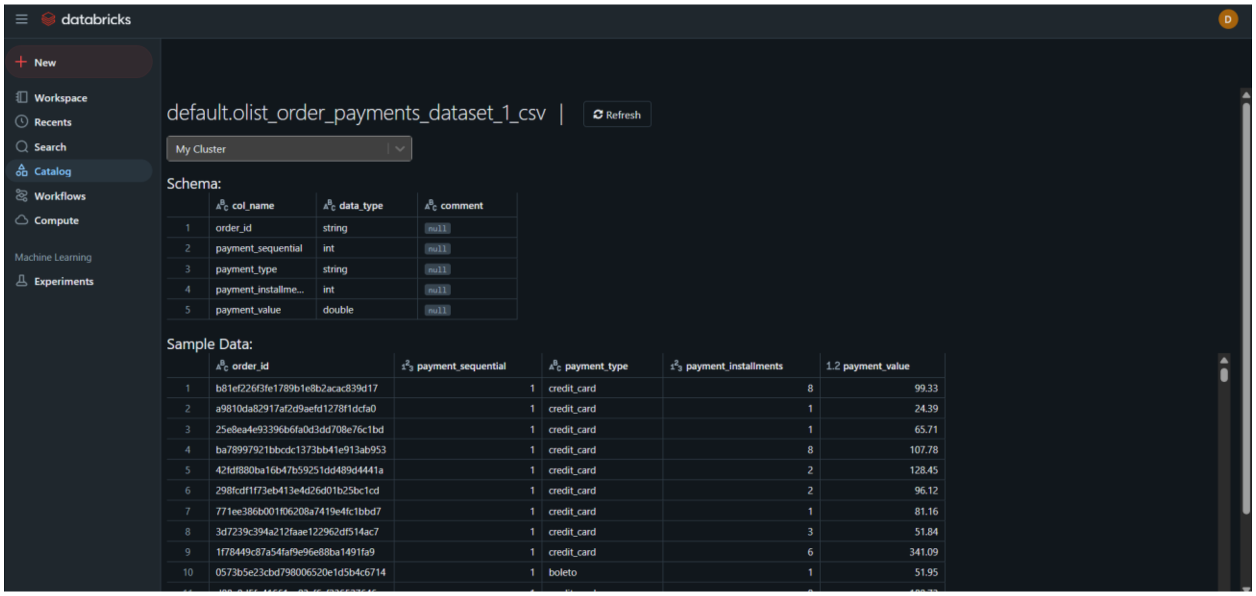
- [x] First row is header
- [x] Infer schema
- [ ] Multi-line

Table Preview

order_id	payment_sequential	payment_type	payment_installments	payment_value
STRING	INT	STRING	INT	DOUBLE
b81ef226f3fe1789b1e8b2acac839d1		credit_card	8	99.33
a9810da82917af2d9aefd1278f1dcf:1		credit_card	1	24.39
25e8ea4e93396b6fa0d3dd708e76c1		credit_card	1	65.71
ba78997921bbcdc1373bb41e913al1		credit_card	8	107.78
42fdf880ba16b47b59251dd489d441		credit_card	2	128.45
298fcdf1f73eb413e4d26d01b25bc11		credit_card	2	96.12
771ee386b001f06208a7419e4fc1bl1		credit_card	1	81.16

DATA WAREHOUSE IMPLEMENTATION PROJECT

Order Payments:



default.olist_order_payments_dataset_1_csv

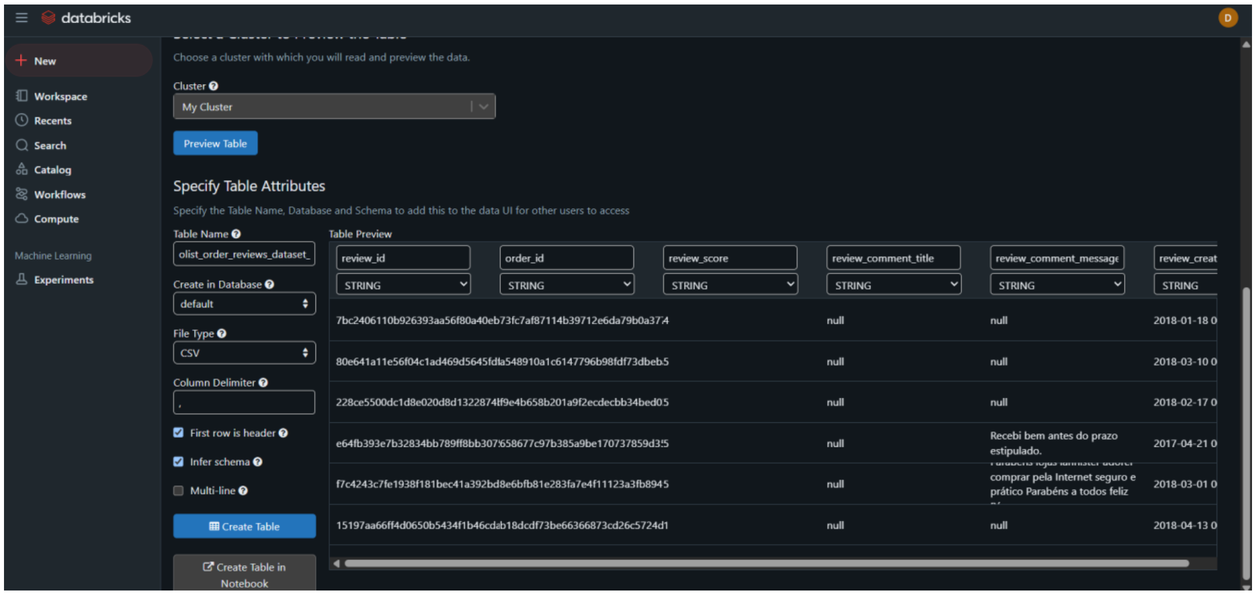
My Cluster

Schema:

	col_name	data_type	comment
1	order_id	string	null
2	payment_sequential	int	null
3	payment_type	string	null
4	payment_installme...	int	null
5	payment_value	double	null

Sample Data:

	order_id	payment_sequential	payment_type	payment_installments	1.2 payment_value
1	b81ef226f3fe1789b1e8b2acac839d17	1	credit_card	8	99.33
2	a9810da82917af2d9aefd1278f1dcfa0	1	credit_card	1	24.39
3	25e8ea4e93396b6fa0d3dd708e76c1bd	1	credit_card	1	65.71
4	ba78997921bbcdc1373bb41e913ab953	1	credit_card	8	107.78
5	42fdf880ba16b47b59251dd489d4441a	1	credit_card	2	128.45
6	298fcdf1f73eb413e4d26d01b25bc1cd	1	credit_card	2	96.12
7	771ee386b001f06208a7419e4fc1bbd7	1	credit_card	1	81.16
8	3d7239c394a212faae122962df514ac7	1	credit_card	3	51.84
9	1f78449c87a54faf9e96e88ba1491fa9	1	credit_card	6	341.09
10	0573b5e23cbd798006520e1d5b4c6714	1	boleto	1	51.95



Choose a cluster with which you will read and preview the data.

Cluster: My Cluster

Specify Table Attributes

Specify the Table Name, Database and Schema to add this to the data UI for other users to access

Table Name: olist_order_reviews_dataset_

Create in Database: default

File Type: CSV

Column Delimiter: ,

[✓] First row is header

[✓] Infer schema

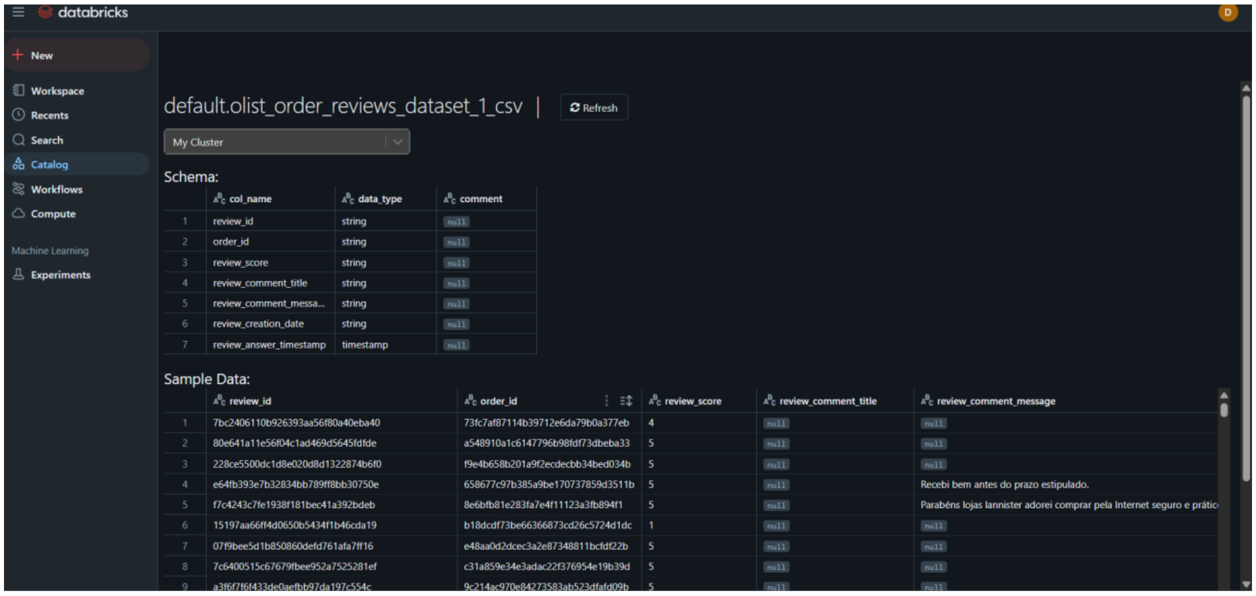
[ ] Multi-line

Table Preview

review_id	order_id	review_score	review_comment_title	review_comment_message	review_creat
STRING	STRING	STRING	STRING	STRING	STRING
7bc2406110b926393aa56f80a40eb73fc7af87114b39712e6da79b0a37:4			null	null	2018-01-18 0
80e641a11e56f04c1ad469d5645fdla548910a1c6147796b98fdf73dbeb5			null	null	2018-03-10 0
228ce5500dc1d8e020d8d1322874lf9e4b658b201a9f2ecdecbb34bed05			null	null	2018-02-17 0
e64fb393e7b32834bb789ff8bb307658677c97b385a9be170737859d3:5			null	Recebi bem antes do prazo estipulado.	2017-04-21 0
f7c4243c7fe1938f181bec41a392bd8e6bfb81e283fa7e4f11123a3fb8945			null	comprar pela Internet seguro e prático Parabéns a todos feliz	2018-03-01 0
15197aa66ff4d0650b5434f1b46cdab18dcdf73be66366873cd26c5724d1			null	null	2018-04-13 0

DATA WAREHOUSE IMPLEMENTATION PROJECT

Order reviews



default.olist_order_reviews_dataset_1_csv

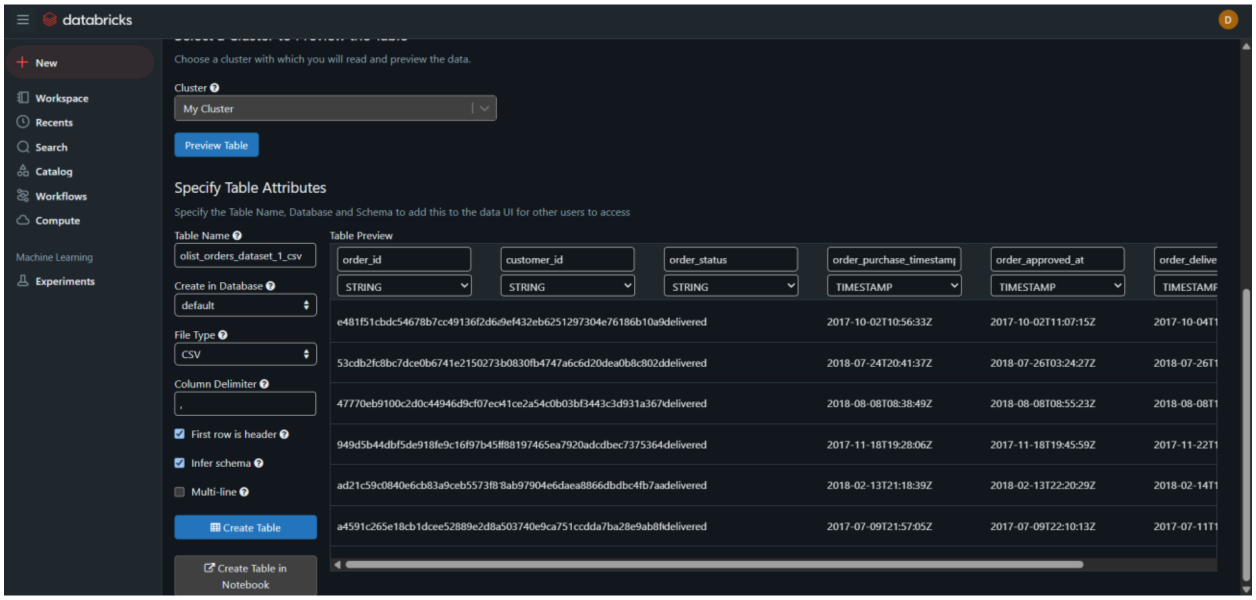
My Cluster

Schema:

	col_name	data_type	comment
1	review_id	string	null
2	order_id	string	null
3	review_score	string	null
4	review_comment_title	string	null
5	review_comment_messa...	string	null
6	review_creation_date	string	null
7	review_answer_timestamp	timestamp	null

Sample Data:

	review_id	order_id	review_score	review_comment_title	review_comment_message
1	7bc2406110b926393aa56f80a40eba40	73fc7af87114b39712e6da79b0a377eb	4	null	null
2	80e641a11e56f04c1ad469d5645fdfde	a548910a1c6147796b98fdf73dbeba33	5	null	null
3	228ce5500dc1d8e020d8d1322874b6f0	f9e4b658b201a9f2ecdecbb34bed034b	5	null	null
4	e64fb393e7b32834bb789ff8bb30750e	658677c97b385a9be170737859d3511b	5	null	Recebi bem antes do prazo estipulado.
5	f7c4243c7fe1938f181bec41a392bdeb	8e6bfb81e283fa7e4f11123a3fb894f1	5	null	Parabéns lojas lannister adorei comprar pela Internet seguro e prátic
6	15197aa66ff4d0650b5434f1b46cda19	b18dcdf73be66366873cd26c5724d1dc	1	null	null
7	07f9bee5d1b850860defd761afa7ff16	e48aa0d2dcec3a2e87348811bcfdf22b	5	null	null
8	7c6400515c67679fbee952a7525281ef	c31a859e34e3adac22f376954e19b39d	5	null	null
9	a3f6f7f6f433de0aefbb97da197c554c	9c214ac970e84273583ab523dfafd09b	5	null	null



Choose a cluster with which you will read and preview the data.

Cluster: My Cluster

Specify Table Attributes

Specify the Table Name, Database and Schema to add this to the data UI for other users to access

Table Name: olist_orders_dataset_1_csv

Create in Database: default

File Type: CSV

Column Delimiter: ,

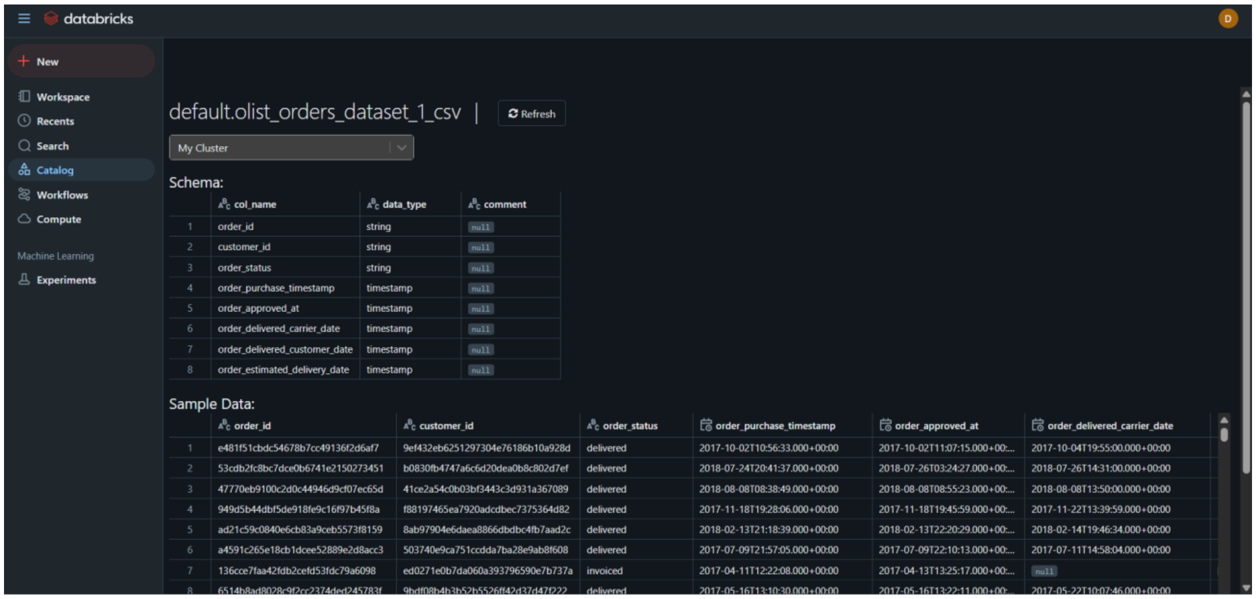
- [x] First row is header
- [x] Infer schema
- [ ] Multi-line

Table Preview

order_id	customer_id	order_status	order_purchase_timestam	order_approved_at	order_delive
STRING	STRING	STRING	TIMESTAMP	TIMESTAMP	TIMESTAMF
e481f51cbdc54678b7cc49136f2d6af9ef432eb6251297304e76186b10a9delivered			2017-10-02T10:56:33Z	2017-10-02T11:07:15Z	2017-10-04T1
53cdb2fc8bc7dce0b6741e2150273b0830fb4747a6c6d20dea0b8c802ddelivered			2018-07-24T20:41:37Z	2018-07-26T03:24:27Z	2018-07-26T1
47770eb9100c2d0c44946d9cf07ec41ce2a54c0b03bf3443c3d931a367delivered			2018-08-08T08:38:49Z	2018-08-08T08:55:23Z	2018-08-08T1
949d5b44dbf5de918fe9c16f97b45ff88197465ea7920adcdbec7375364delivered			2017-11-18T19:28:06Z	2017-11-18T19:45:59Z	2017-11-22T1
ad21c59c0840e6cb83a9ceb5573f8 8ab97904e6daea8866dbdbc4fb7aadelivered			2018-02-13T21:18:39Z	2018-02-13T22:20:29Z	2018-02-14T1
a4591c265e18cb1dcee52889e2d8a503740e9ca751ccdda7ba28e9ab8fdelivered			2017-07-09T21:57:05Z	2017-07-09T22:10:13Z	2017-07-11T1

DATA WAREHOUSE IMPLEMENTATION PROJECT

Orders:



default.olist_orders_dataset_1_csv

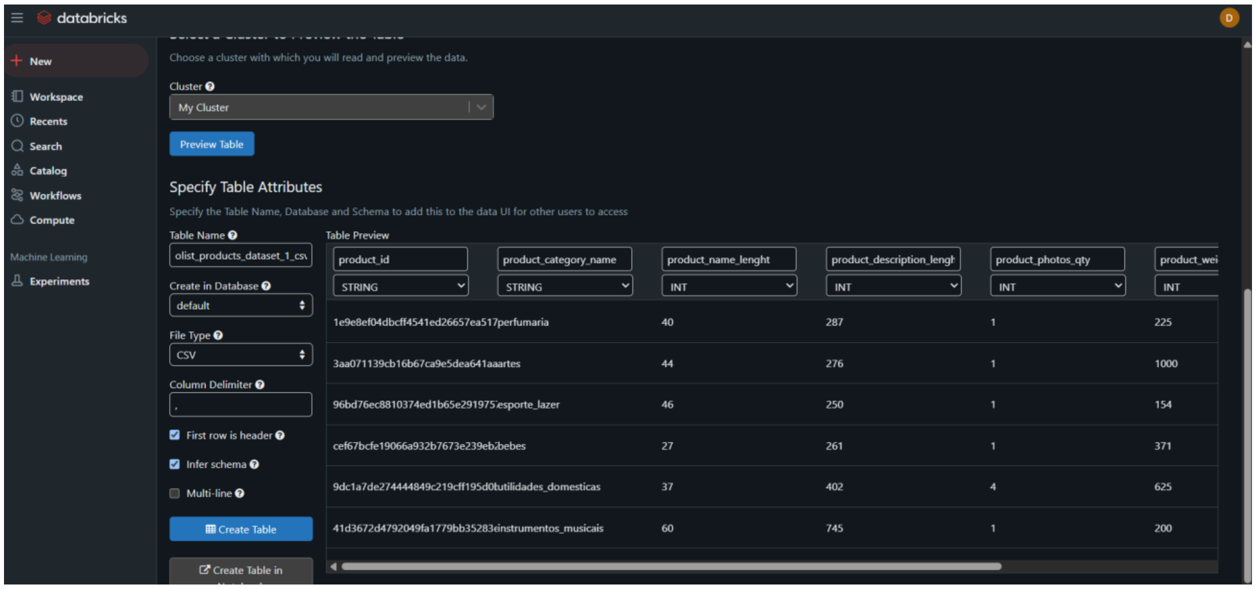
My Cluster

Schema:

	col_name	data_type	comment
1	order_id	string	null
2	customer_id	string	null
3	order_status	string	null
4	order_purchase_timestamp	timestamp	null
5	order_approved_at	timestamp	null
6	order_delivered_carrier_date	timestamp	null
7	order_delivered_customer_date	timestamp	null
8	order_estimated_delivery_date	timestamp	null

Sample Data:

	order_id	customer_id	order_status	order_purchase_timestamp	order_approved_at	order_delivered_carrier_date
1	e481f51cbdc54678b7cc49136f2d6af7	9ef432eb6251297304e76186b10a928d	delivered	2017-10-02T10:56:33.000+00:00	2017-10-02T11:07:15.000+00:...	2017-10-04T19:55:00.000+00:00
2	53cdb2fc8bc7dce0b6741e2150273451	b0830fb4747a6c6d20dea0b8c802d7ef	delivered	2018-07-24T20:41:37.000+00:00	2018-07-26T03:24:27.000+00:...	2018-07-26T14:31:00.000+00:00
3	47770eb9100c2d0c44946d9cf07ec65d	41ce2a54c0b03bf3443c3d931a367089	delivered	2018-08-08T08:38:49.000+00:00	2018-08-08T08:55:23.000+00:...	2018-08-08T13:50:00.000+00:00
4	949d5b44dbf5de918fe9c16f97b45f8a	f88197465ea7920adcdbec7375364d82	delivered	2017-11-18T19:28:06.000+00:00	2017-11-18T19:45:59.000+00:...	2017-11-22T13:39:59.000+00:00
5	ad21c59c0840e6cb83a9ceb5573f8159	8ab97904e6daea8866dbdbc4fb7aad2c	delivered	2018-02-13T21:18:39.000+00:00	2018-02-13T22:20:29.000+00:...	2018-02-14T19:46:34.000+00:00
6	a4591c265e18cb1dcee52889e2d8acc3	503740e9ca751ccdda7ba28e9ab8f608	delivered	2017-07-09T21:57:05.000+00:00	2017-07-09T22:10:13.000+00:...	2017-07-11T14:58:04.000+00:00
7	136cce7faa42fdb2cefd53fdc79a6098	ed0271e0b7da060a393796590e7b737a	invoiced	2017-04-11T12:22:08.000+00:00	2017-04-13T13:25:17.000+00:...	null
8	6514b8ad8028c9f2cc2374ded245783f	9bdf08b4b3b52b5526ff42d37d47f222	delivered	2017-05-16T13:10:30.000+00:00	2017-05-16T13:22:11.000+00:...	2017-05-22T10:07:46.000+00:00



Choose a cluster with which you will read and preview the data.

Cluster: My Cluster

Preview Table

Specify Table Attributes

Specify the Table Name, Database and Schema to add this to the data UI for other users to access

Table Name: olist_products_dataset_1_csv

Create in Database: default

File Type: CSV

Column Delimiter: ,

- [x] First row is header
- [x] Infer schema
- [ ] Multi-line

Create Table

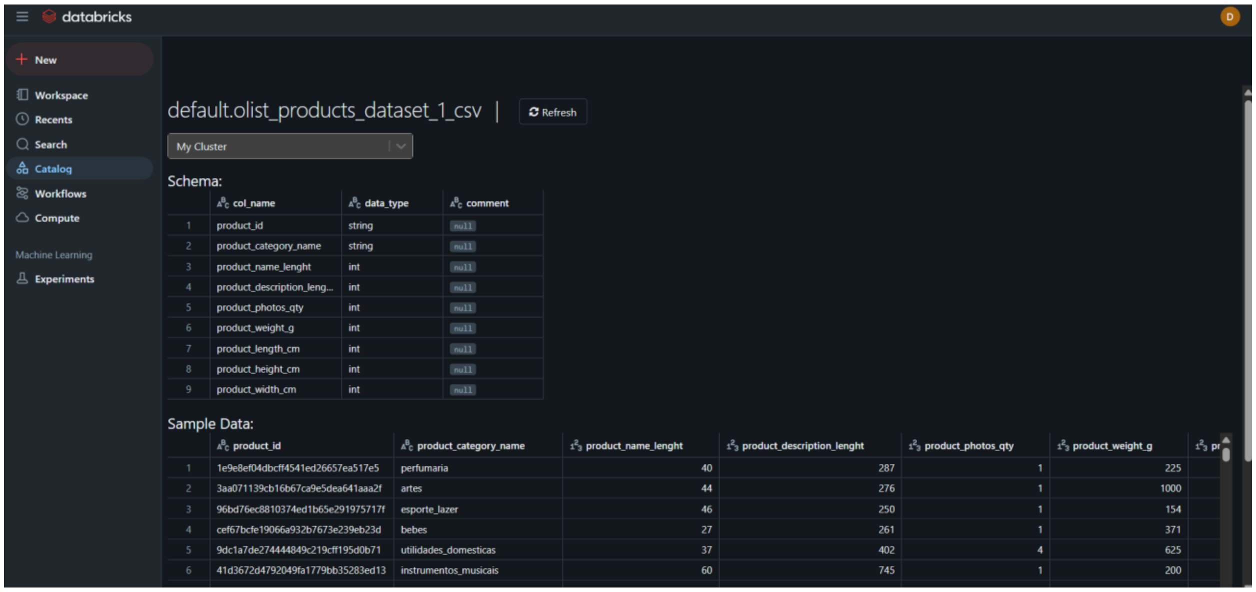
Create Table in Notebook

Table Preview

product_id	product_category_name	product_name_lenght	product_description_lenght	product_photos_qty	product_wei
STRING	STRING	INT	INT	INT	INT
1e9e8ef04dbcff4541ed26657ea517	perfumaria	40	287	1	225
3aa071139cb16b67ca9e5dea641a	artes	44	276	1	1000
96bd76ec8810374ed1b65e291975	esporte_lazer	46	250	1	154
cef67bcfe19066a932b7673e239eb	bebes	27	261	1	371
9dc1a7de274444849c219cff195d0	utilidades_domesticas	37	402	4	625
41d3672d4792049fa1779bb35283	instrumentos_musicais	60	745	1	200

DATA WAREHOUSE IMPLEMENTATION PROJECT

Products:



default.olist_products_dataset_1_csv

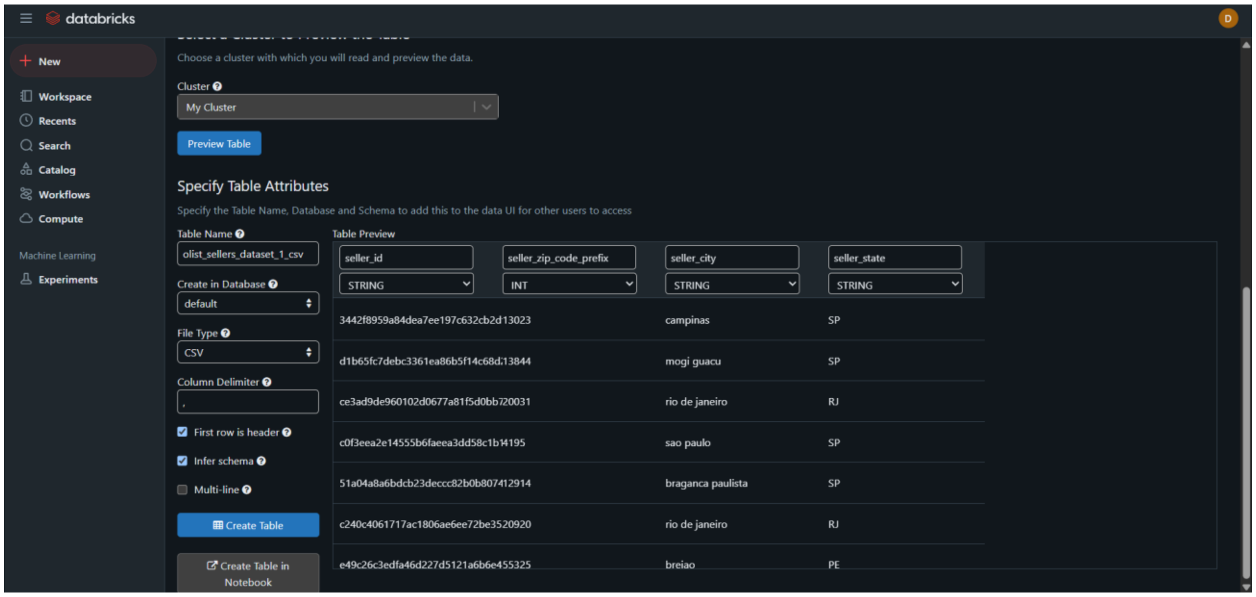
My Cluster

Schema:

	col_name	data_type	comment
1	product_id	string	null
2	product_category_name	string	null
3	product_name_lenght	int	null
4	product_description_leng...	int	null
5	product_photos_qty	int	null
6	product_weight_g	int	null
7	product_length_cm	int	null
8	product_height_cm	int	null
9	product_width_cm	int	null

Sample Data:

	product_id	product_category_name	product_name_lenght	product_description_lenght	product_photos_qty	product_weight_g	pr
1	1e9e8ef04dbcff4541ed26657ea517e5	perfumaria	40	287	1	225	
2	3aa071139cb16b67ca9e5dea641aaa2f	artes	44	276	1	1000	
3	96bd76ec8810374ed1b65e291975717f	esporte_lazer	46	250	1	154	
4	cef67bcfe19066a932b7673e239eb23d	bebes	27	261	1	371	
5	9dc1a7de274444849c219cff195d0b71	utilidades_domesticas	37	402	4	625	
6	41d3672d4792049fa1779bb35283ed13	instrumentos_musicais	60	745	1	200	



Choose a cluster with which you will read and preview the data.

Cluster: My Cluster

Specify Table Attributes

Specify the Table Name, Database and Schema to add this to the data UI for other users to access

Table Name: olist_sellers_dataset_1_csv

Create in Database: default

File Type: CSV

Column Delimiter: ,

[✓] First row is header

[✓] Infer schema

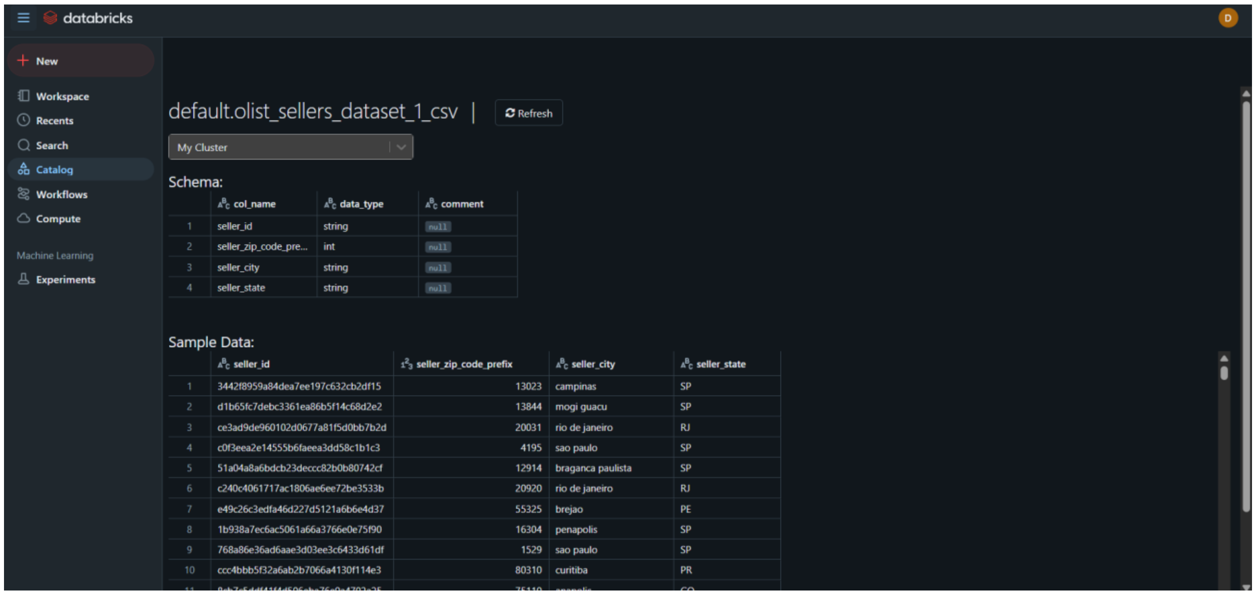
[ ] Multi-line

Table Preview

seller_id	seller_zip_code_prefix	seller_city	seller_state
STRING	INT	STRING	STRING
3442f8959a84dea7ee197c632cb2d13023		campinas	SP
d1b65fc7debc3361ea86b5f14c68d13844		mogi guacu	SP
ce3ad9de960102d0677a81f5d0bb720031		rio de janeiro	RJ
c0f3eea2e14555b6faeea3dd58c1b14195		sao paulo	SP
51a04a8a6bdcb23deccc82b0b807412914		braganca paulista	SP
c240c4061717ac1806ae6ee72be3520920		rio de janeiro	RJ
e49c26c3edfa46d227d5121a6b6e455325		breiao	PE

DATA WAREHOUSE IMPLEMENTATION PROJECT

Sellers:



default.olist_sellers_dataset_1_csv

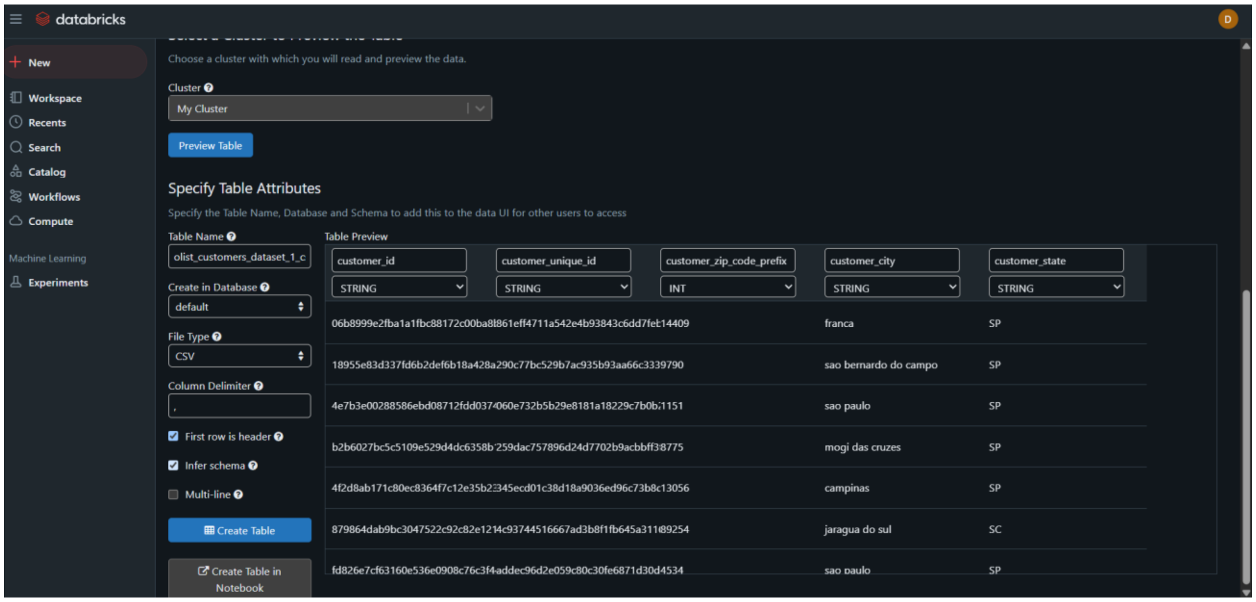
My Cluster

Schema:

	col_name	data_type	comment
1	seller_id	string	null
2	seller_zip_code_pre...	int	null
3	seller_city	string	null
4	seller_state	string	null

Sample Data:

	seller_id	seller_zip_code_prefix	seller_city	seller_state
1	3442f8959a84dea7ee197c632cb2df15	13023	campinas	SP
2	d1b65fc7debc3361ea86b5f14c68d2e2	13844	mogi guacu	SP
3	ce3ad9de960102d0677a81f5d0bb7b2d	20031	rio de janeiro	RJ
4	c0f3eea2e14555b6faeea3dd58c1b1c3	4195	sao paulo	SP
5	51a04a8a6bdcb23deccc82b0b80742cf	12914	braganca paulista	SP
6	c240c4061717ac1806ae6ee72be3533b	20920	rio de janeiro	RJ
7	e49c26c3edfa46d227d5121a6b6e4d37	55325	brejao	PE
8	1b938a7ec6ac5061a66a3766e0e75f90	16304	penapolis	SP
9	768a86e36ad6aae3d03ee3c6433d61df	1529	sao paulo	SP
10	ccc4bbb5f32a6ab2b7066a4130f114e3	80310	curitiba	PR



Choose a cluster with which you will read and preview the data.

Cluster: My Cluster

Preview Table

Specify Table Attributes

Specify the Table Name, Database and Schema to add this to the data UI for other users to access

Table Name: olist_customers_dataset_1_c

Create in Database: default

File Type: CSV

Column Delimiter: ,

[✓] First row is header

[✓] Infer schema

[ ] Multi-line

Create Table

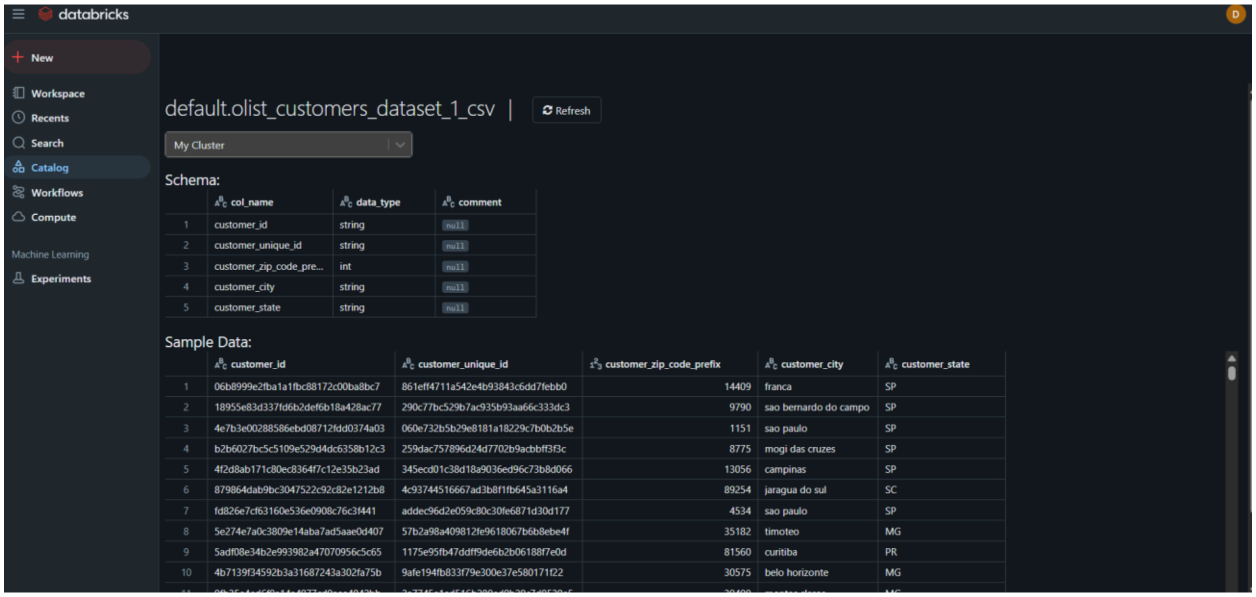
Create Table in Notebook

Table Preview

customer_id	customer_unique_id	customer_zip_code_prefix	customer_city	customer_state
STRING	STRING	INT	STRING	STRING
06b8999e2fba1a1fbc88172c00ba8l861eff4711a542e4b93843c6dd7fet14409			franca	SP
18955e83d337fd6b2def6b18a428a290c77bc529b7ac935b93aa66c3339790			sao bernardo do campo	SP
4e7b3e00288586ebd08712fdd0374060e732b5b29e8181a18229c7b0b;1151			sao paulo	SP
b2b6027bc5c5109e529d4dc6358b'259dac757896d24d7702b9acbbff38775			mogi das cruzes	SP
4f2d8ab171c80ec8364f7c12e35b2345ecd01c38d18a9036ed96c73b8c13056			campinas	SP
879864dab9bc3047522c92c82e1214c93744516667ad3b8f1fb645a31189254			jaragua do sul	SC
fd826e7cf63160e536e0908c76c3f4addec96d2e059c80c30fe6871d30d4534			sao paulo	SP

DATA WAREHOUSE IMPLEMENTATION PROJECT

Customers:



default.olist_customers_dataset_1_csv

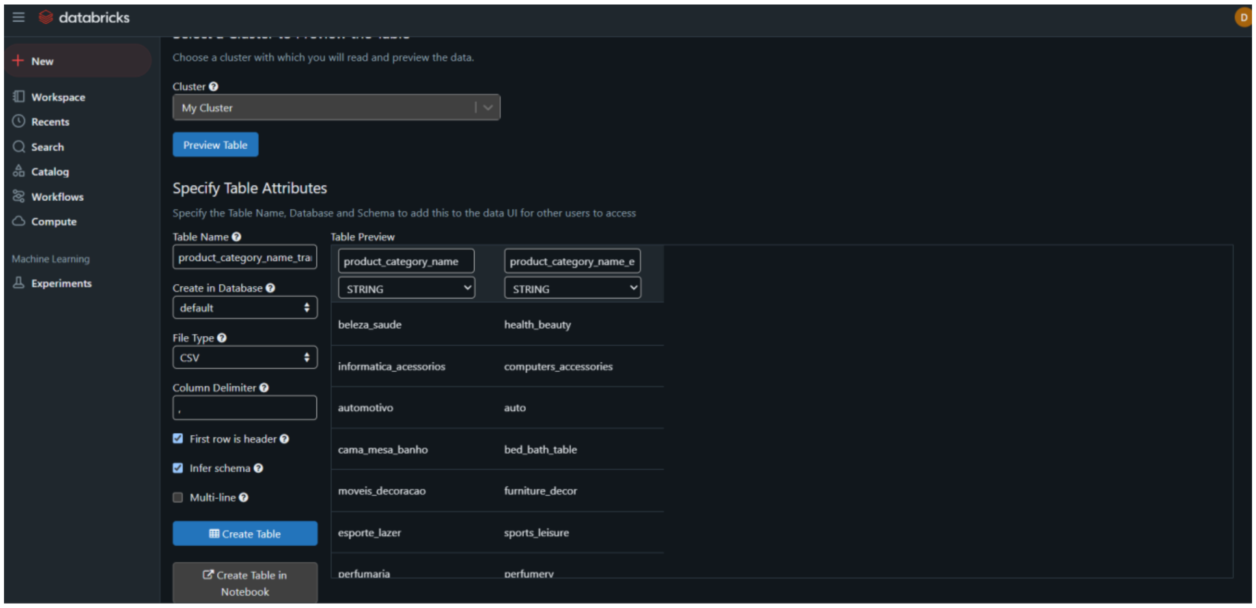
My Cluster

Schema:

	col_name	data_type	comment
1	customer_id	string	null
2	customer_unique_id	string	null
3	customer_zip_code_pre...	int	null
4	customer_city	string	null
5	customer_state	string	null

Sample Data:

	customer_id	customer_unique_id	customer_zip_code_prefix	customer_city	customer_state
1	06b8999e2fba1a1fbc88172c00ba8bc7	861eff4711a542e4b93843c6dd7febb0	14409	franca	SP
2	18955e83d337fd6b2def6b18a428ac77	290c77bc529b7ac935b93aa66c333dc3	9790	sao bernardo do campo	SP
3	4e7b3e00288586ebd08712fdd0374a03	060e732b5b29e8181a18229c7b0b2b5e	1151	sao paulo	SP
4	b2b6027bc5c5109e529d4dc6358b12c3	259dac757896d24d7702b9acbbff3f3c	8775	mogi das cruzes	SP
5	4f2d8ab171c80ec8364f7c12e35b23ad	345ecd01c38d18a9036ed96c73b8d066	13056	campinas	SP
6	879864dab9bc3047522c92c82e1212b8	4c93744516667ad3b8f1fb645a3116a4	89254	jaragua do sul	SC
7	fd826e7cf63160e536e0908c76c3f441	addec96d2e059c80c30fe6871d30d177	4534	sao paulo	SP
8	5e274e7a0c3809e14aba7ad5aae0d407	57b2a98a409812fe9618067b6b8ebe4f	35182	timoteo	MG
9	5adf08e34b2e993982a47070956c5c65	1175e95fb47ddff9de6b2b06188f7e0d	81560	curitiba	PR
10	4b7139f34592b3a31687243a302fa75b	9afe194fb833f79e300e37e580171f22	30575	belo horizonte	MG



Choose a cluster with which you will read and preview the data.

Cluster: My Cluster

Specify Table Attributes

Specify the Table Name, Database and Schema to add this to the data UI for other users to access

Table Name: product_category_name_tra

Create in Database: default

File Type: CSV

Column Delimiter: ,

☑ First row is header

☑ Infer schema

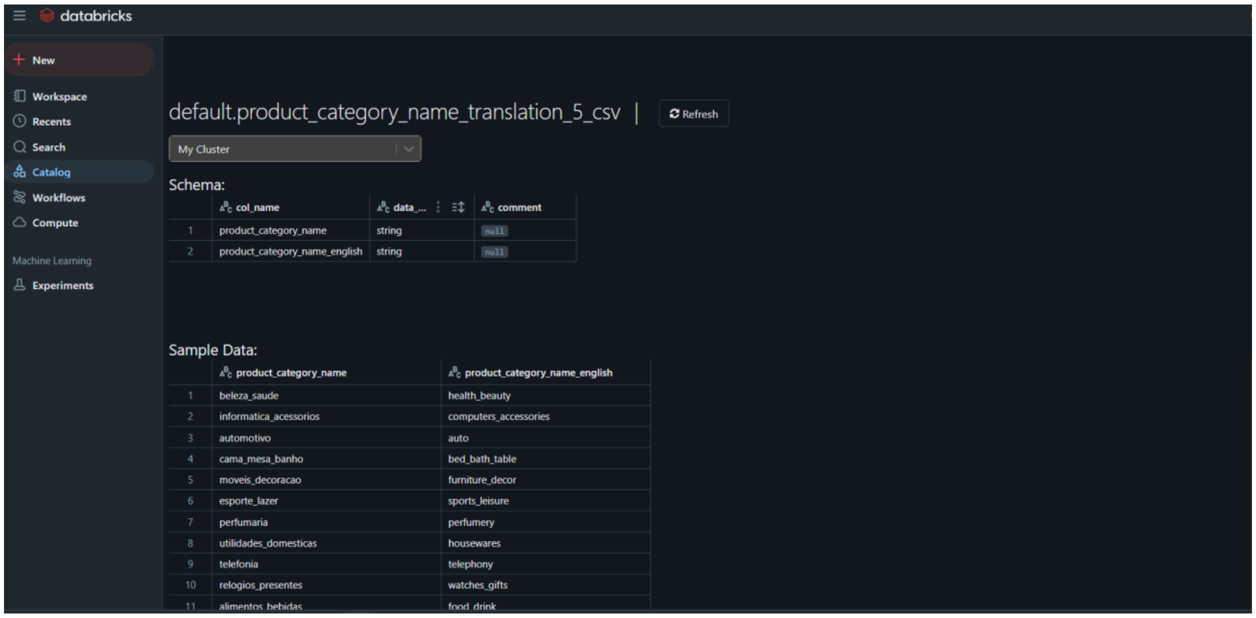
☐ Multi-line

Table Preview

product_category_name	product_category_name_e
STRING	STRING
beleza_saude	health_beauty
informatica_acessorios	computers_accessories
automotivo	auto
cama_mesa_banho	bed_bath_table
moveis_decoracao	furniture_decor
esporte_lazer	sports_leisure
perfumaria	perfumery

DATA WAREHOUSE IMPLEMENTATION PROJECT

Translation:



default.product_category_name_translation_5_csv

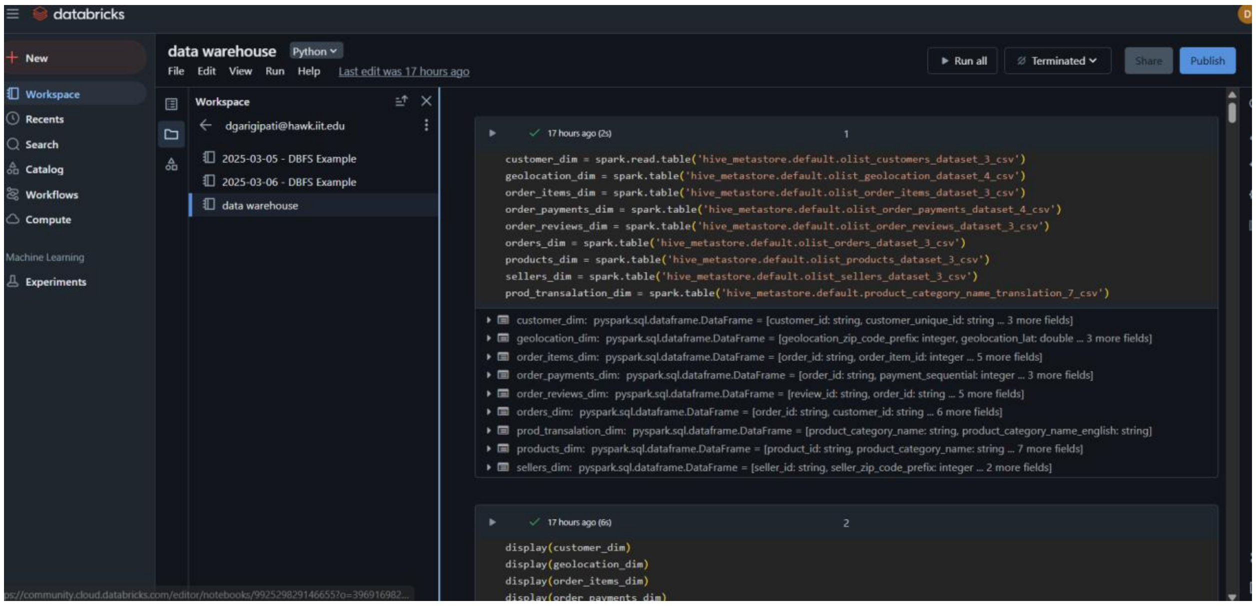
Schema:

	col_name	data_...	comment
1	product_category_name	string	null
2	product_category_name_english	string	null

Sample Data:

	product_category_name	product_category_name_english
1	beleza_saude	health_beauty
2	informatica_acessorios	computers_accessories
3	automotivo	auto
4	cama_mesa_banho	bed_bath_table
5	moveis_decoracao	furniture_decor
6	esporte_lazer	sports_leisure
7	perfumaria	perfumery
8	utilidades_domesticas	housewares
9	telefonia	telephony
10	relogios_presentes	watches_gifts
11	alimentos_bebidas	food_drink

Loading the data into dimensions:



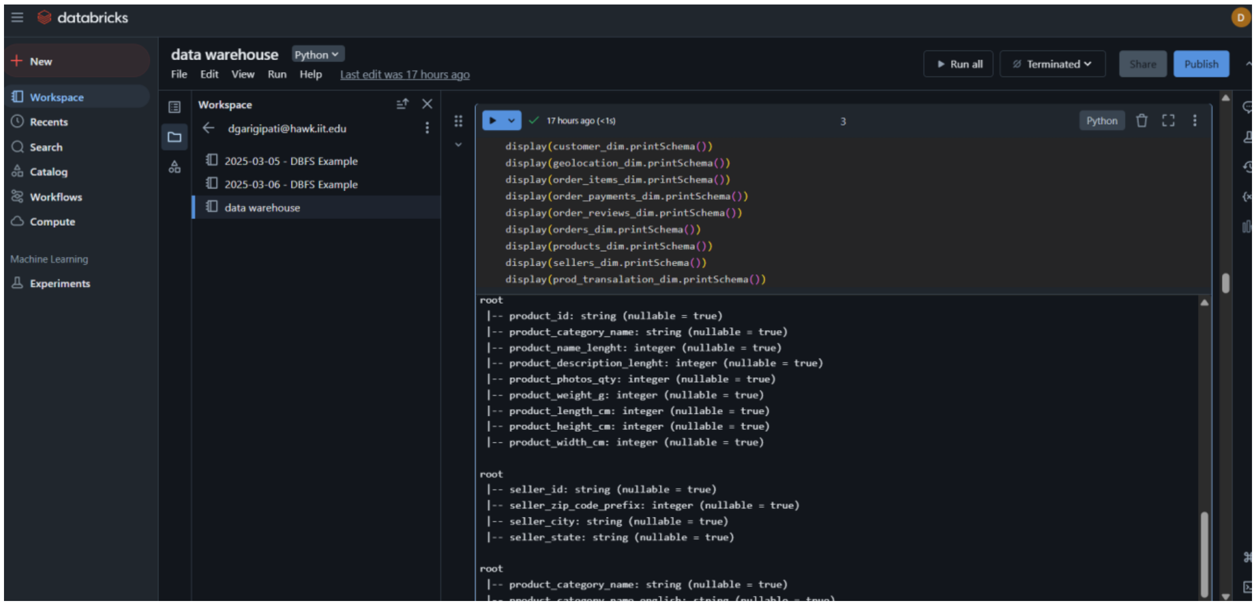
```
customer_dim = spark.read.table('hive_metastore.default.olist_customers_dataset_3_csv')
geolocation_dim = spark.table('hive_metastore.default.olist_geolocation_dataset_4_csv')
order_items_dim = spark.table('hive_metastore.default.olist_order_items_dataset_3_csv')
order_payments_dim = spark.table('hive_metastore.default.olist_order_payments_dataset_4_csv')
order_reviews_dim = spark.table('hive_metastore.default.olist_order_reviews_dataset_3_csv')
orders_dim = spark.table('hive_metastore.default.olist_orders_dataset_3_csv')
products_dim = spark.table('hive_metastore.default.olist_products_dataset_3_csv')
sellers_dim = spark.table('hive_metastore.default.olist_sellers_dataset_3_csv')
prod_transalation_dim = spark.table('hive_metastore.default.product_category_name_translation_7_csv')
```

```
customer_dim: pyspark.sql.dataframe.DataFrame = [customer_id: string, customer_unique_id: string ... 3 more fields]
geolocation_dim: pyspark.sql.dataframe.DataFrame = [geolocation_zip_code_prefix: integer, geolocation_lat: double ... 3 more fields]
order_items_dim: pyspark.sql.dataframe.DataFrame = [order_id: string, order_item_id: integer ... 5 more fields]
order_payments_dim: pyspark.sql.dataframe.DataFrame = [order_id: string, payment_sequential: integer ... 3 more fields]
order_reviews_dim: pyspark.sql.dataframe.DataFrame = [review_id: string, order_id: string ... 5 more fields]
orders_dim: pyspark.sql.dataframe.DataFrame = [order_id: string, customer_id: string ... 6 more fields]
prod_transalation_dim: pyspark.sql.dataframe.DataFrame = [product_category_name: string, product_category_name_english: string]
products_dim: pyspark.sql.dataframe.DataFrame = [product_id: string, product_category_name: string ... 7 more fields]
sellers_dim: pyspark.sql.dataframe.DataFrame = [seller_id: string, seller_zip_code_prefix: integer ... 2 more fields]
```

```
display(customer_dim)
display(geolocation_dim)
display(order_items_dim)
display(order_payments_dim)
```

DATA WAREHOUSE IMPLEMENTATION PROJECT

Displaying Schema of dimension:



```
display(customer_dim.printSchema())
display(geolocation_dim.printSchema())
display(order_items_dim.printSchema())
display(order_payments_dim.printSchema())
display(order_reviews_dim.printSchema())
display(orders_dim.printSchema())
display(products_dim.printSchema())
display(sellers_dim.printSchema())
display(prod_transalation_dim.printSchema())
```

```
root
 |-- product_id: string (nullable = true)
 |-- product_category_name: string (nullable = true)
 |-- product_name_lenght: integer (nullable = true)
 |-- product_description_lenght: integer (nullable = true)
 |-- product_photos_qty: integer (nullable = true)
 |-- product_weight_g: integer (nullable = true)
 |-- product_length_cm: integer (nullable = true)
 |-- product_height_cm: integer (nullable = true)
 |-- product_width_cm: integer (nullable = true)

root
 |-- seller_id: string (nullable = true)
 |-- seller_zip_code_prefix: integer (nullable = true)
 |-- seller_city: string (nullable = true)
 |-- seller_state: string (nullable = true)

root
 |-- product_category_name: string (nullable = true)
```

Customer_dim
root
 |-- customer_id: string (nullable = true)
 |-- customer_unique_id: string (nullable = true)
 |-- customer_zip_code_prefix: integer (nullable = true)
 |-- customer_city: string (nullable = true)
 |-- customer_state: string (nullable = true)

Geolocation_dim
root
 |-- geolocation_zip_code_prefix: integer (nullable = true)
 |-- geolocation_lat: double (nullable = true)
 |-- geolocation_lng: double (nullable = true)
 |-- geolocation_city: string (nullable = true)
 |-- geolocation_state: string (nullable = true)

Order_itmes_dim
root
 |-- order_id: string (nullable = true)
 |-- order_item_id: integer (nullable = true)
 |-- product_id: string (nullable = true)
 |-- seller_id: string (nullable = true)
 |-- shipping_limit_date: timestamp (nullable = true)
 |-- price: double (nullable = true)
 |-- freight_value: double (nullable = true)

DATA WAREHOUSE IMPLEMENTATION PROJECT

Order_payments_dim

```
root
 |-- order_id: string (nullable = true)
 |-- payment_sequential: integer (nullable = true)
 |-- payment_type: string (nullable = true)
 |-- payment_installments: integer (nullable = true)
 |-- payment_value: double (nullable = true)
```

Order_reviews_dim

```
root
 |-- review_id: string (nullable = true)
 |-- order_id: string (nullable = true)
 |-- review_score: string (nullable = true)
 |-- review_comment_title: string (nullable = true)
 |-- review_comment_message: string (nullable = true)
 |-- review_creation_date: string (nullable = true)
 |-- review_answer_timestamp: timestamp (nullable = true)
```

Orders_dim

```
root
 |-- order_id: string (nullable = true)
 |-- customer_id: string (nullable = true)
 |-- order_status: string (nullable = true)
 |-- order_purchase_timestamp: timestamp (nullable = true)
 |-- order_approved_at: timestamp (nullable = true)
 |-- order_delivered_carrier_date: timestamp (nullable = true)
 |-- order_delivered_customer_date: timestamp (nullable = true)
 |-- order_estimated_delivery_date: timestamp (nullable = true)
```

Products_dim

```
root
 |-- product_id: string (nullable = true)
 |-- product_category_name: string (nullable = true)
 |-- product_name_lenght: integer (nullable = true)
 |-- product_description_lenght: integer (nullable = true)
 |-- product_photos_qty: integer (nullable = true)
 |-- product_weight_g: integer (nullable = true)
 |-- product_length_cm: integer (nullable = true)
 |-- product_height_cm: integer (nullable = true)
 |-- product_width_cm: integer (nullable = true)
```

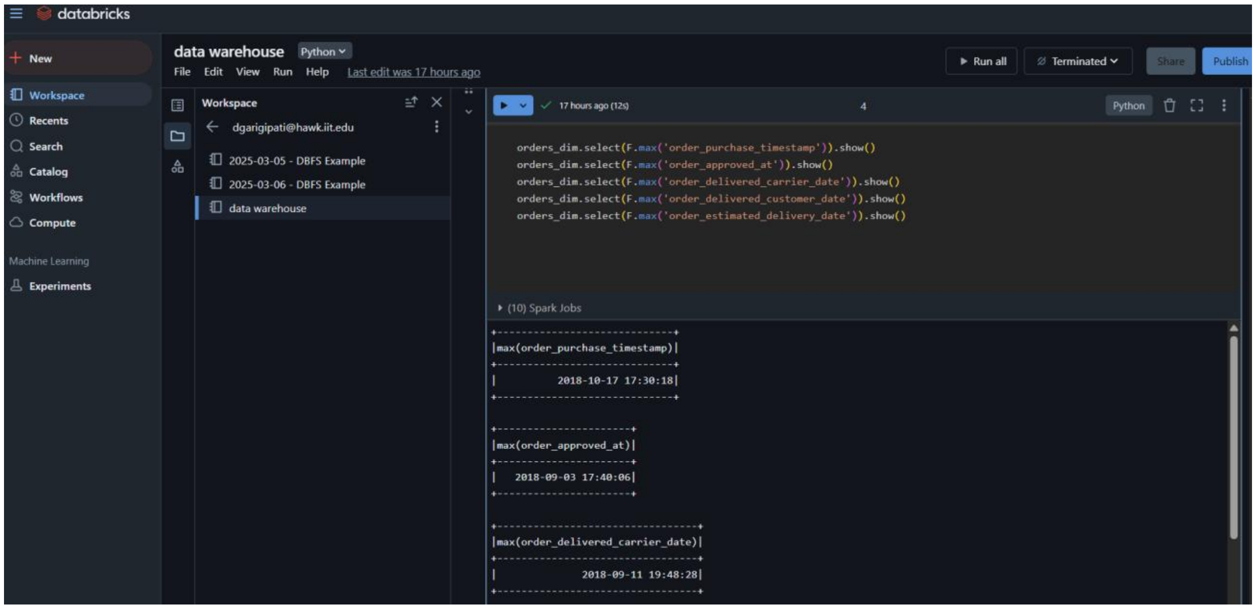
Sellers_dim

```
root
 |-- seller_id: string (nullable = true)
 |-- seller_zip_code_prefix: integer (nullable = true)
 |-- seller_city: string (nullable = true)
 |-- seller_state: string (nullable = true)
```

DATA WAREHOUSE IMPLEMENTATION PROJECT

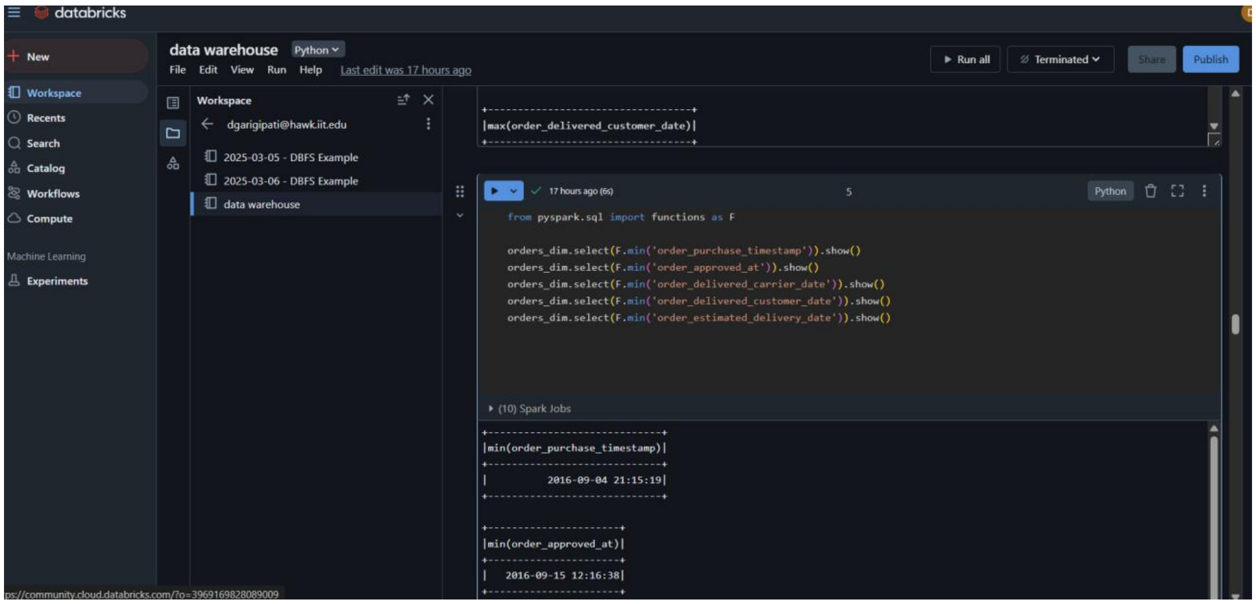
```
Prod_transalation_dim
root
 |-- product_category_name: string (nullable = true)
 |-- product_category_name_english: string (nullable = true)
```

Getting Max and Min dates to generate date dimension:



```
orders_dim.select(F.max('order_purchase_timestamp')).show()
orders_dim.select(F.max('order_approved_at')).show()
orders_dim.select(F.max('order_delivered_carrier_date')).show()
orders_dim.select(F.max('order_delivered_customer_date')).show()
orders_dim.select(F.max('order_estimated_delivery_date')).show()
```

```
+-----------------------------+
|max(order_purchase_timestamp)|
+-----------------------------+
|          2018-10-17 17:30:18|
+-----------------------------+

+----------------------+
|max(order_approved_at)|
+----------------------+
|   2018-09-03 17:40:06|
+----------------------+

+---------------------------------+
|max(order_delivered_carrier_date)|
+---------------------------------+
|              2018-09-11 19:48:28|
+---------------------------------+
```



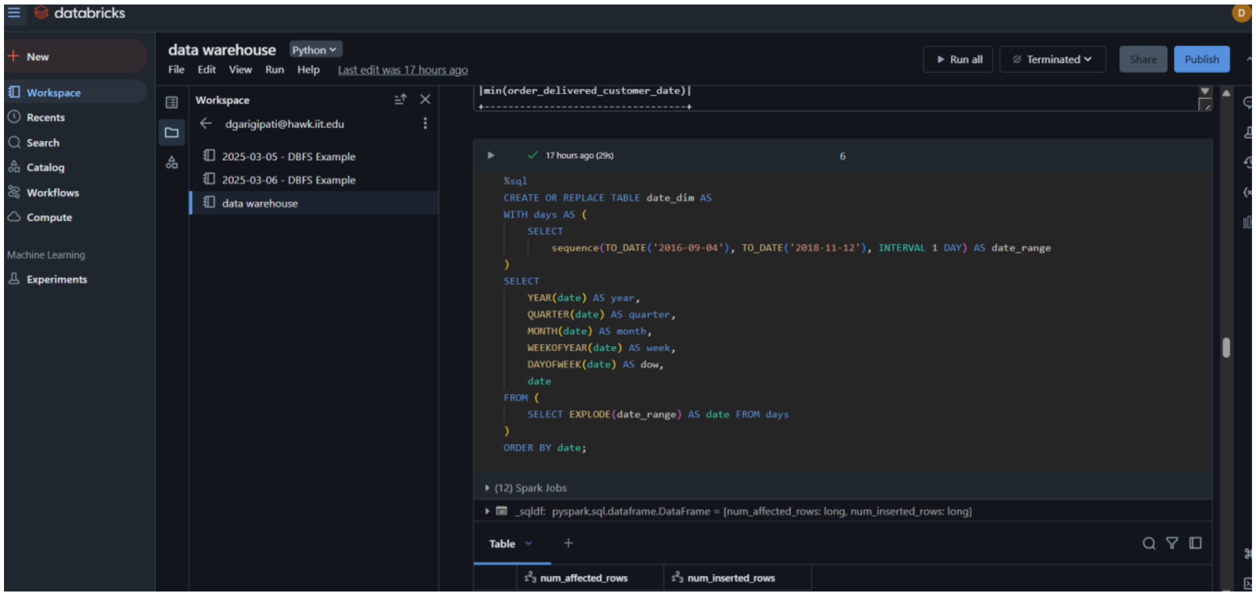
```
+----------------------------------+
|max(order_delivered_customer_date)|
+----------------------------------+
```

```
from pyspark.sql import functions as F

orders_dim.select(F.min('order_purchase_timestamp')).show()
orders_dim.select(F.min('order_approved_at')).show()
orders_dim.select(F.min('order_delivered_carrier_date')).show()
orders_dim.select(F.min('order_delivered_customer_date')).show()
orders_dim.select(F.min('order_estimated_delivery_date')).show()
```

```
+-----------------------------+
|min(order_purchase_timestamp)|
+-----------------------------+
|          2016-09-04 21:15:19|
+-----------------------------+

+----------------------+
|min(order_approved_at)|
+----------------------+
|   2016-09-15 12:16:38|
+----------------------+
```

DATA WAREHOUSE IMPLEMENTATION PROJECT

Generating Date Dim:



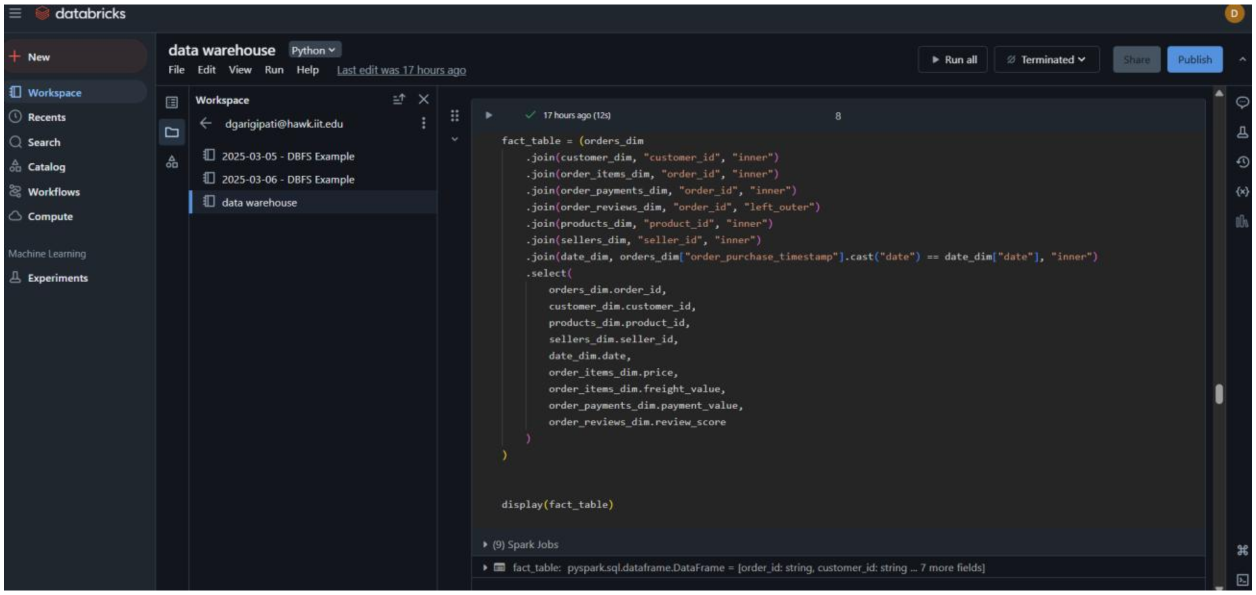
```
|min(order_delivered_customer_date)|
+----------------------------------+
```

```
%sql
CREATE OR REPLACE TABLE date_dim AS
WITH days AS (
    SELECT
        sequence(TO_DATE('2016-09-04'), TO_DATE('2018-11-12'), INTERVAL 1 DAY) AS date_range
)
SELECT
    YEAR(date) AS year,
    QUARTER(date) AS quarter,
    MONTH(date) AS month,
    WEEKOFYEAR(date) AS week,
    DAYOFWEEK(date) AS dow,
    date
FROM (
    SELECT EXPLODE(date_range) AS date FROM days
)
ORDER BY date;
```

```
(12) Spark Jobs
_sqldf: pyspark.sql.dataframe.DataFrame = [num_affected_rows: long, num_inserted_rows: long]
```

num_affected_rows	num_inserted_rows

Creating fact table by joining all the dimensions:

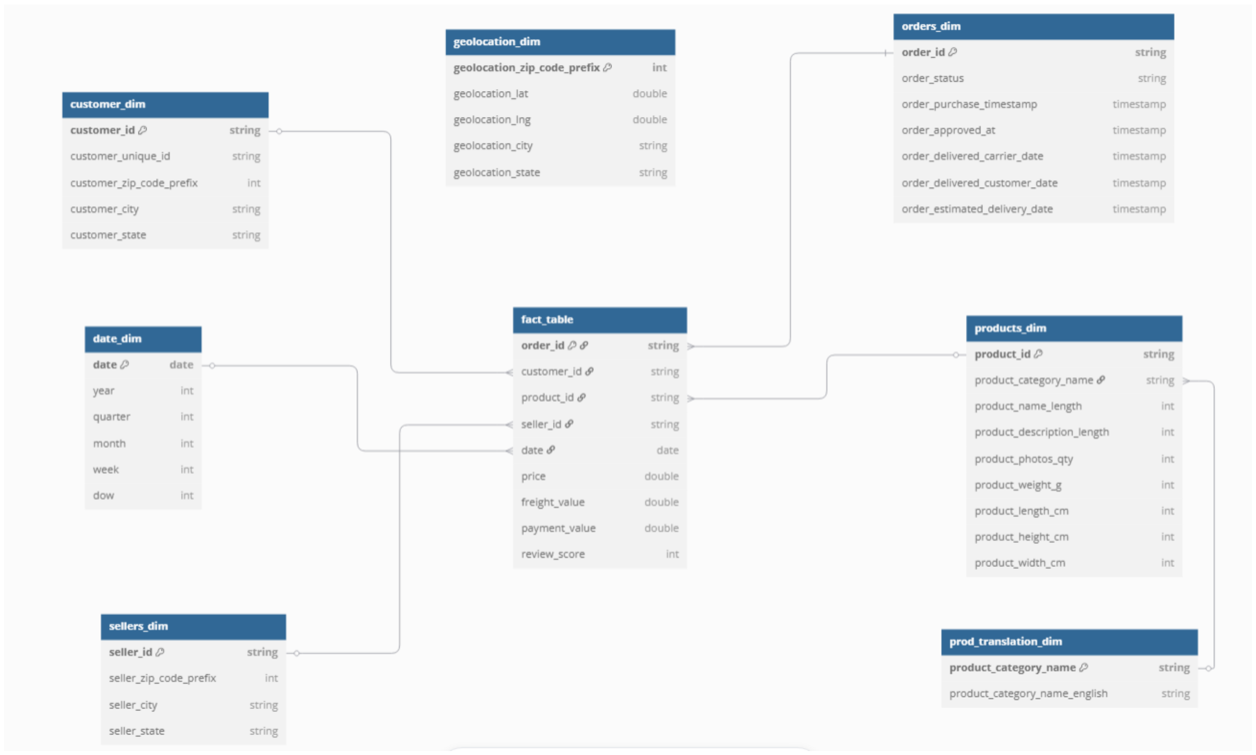


```
fact_table = (orders_dim
    .join(customer_dim, "customer_id", "inner")
    .join(order_items_dim, "order_id", "inner")
    .join(order_payments_dim, "order_id", "inner")
    .join(order_reviews_dim, "order_id", "left_outer")
    .join(products_dim, "product_id", "inner")
    .join(sellers_dim, "seller_id", "inner")
    .join(date_dim, orders_dim["order_purchase_timestamp"].cast("date") == date_dim["date"], "inner")
    .select(
        orders_dim.order_id,
        customer_dim.customer_id,
        products_dim.product_id,
        sellers_dim.seller_id,
        date_dim.date,
        order_items_dim.price,
        order_items_dim.freight_value,
        order_payments_dim.payment_value,
        order_reviews_dim.review_score
    )
)


display(fact_table)
```

```
(9) Spark Jobs
fact_table: pyspark.sql.dataframe.DataFrame = [order_id: string, customer_id: string ... 7 more fields]
```

DATA WAREHOUSE IMPLEMENTATION PROJECT

Fact and dimension ERD in star Schema:



The ERD represents a star schema designed for an e-commerce sales analysis system. At the center is the **fact_table** table, which records transactional data, including order details, prices, freight costs, payment values, and review scores. It is linked to multiple dimension tables that provide descriptive context.

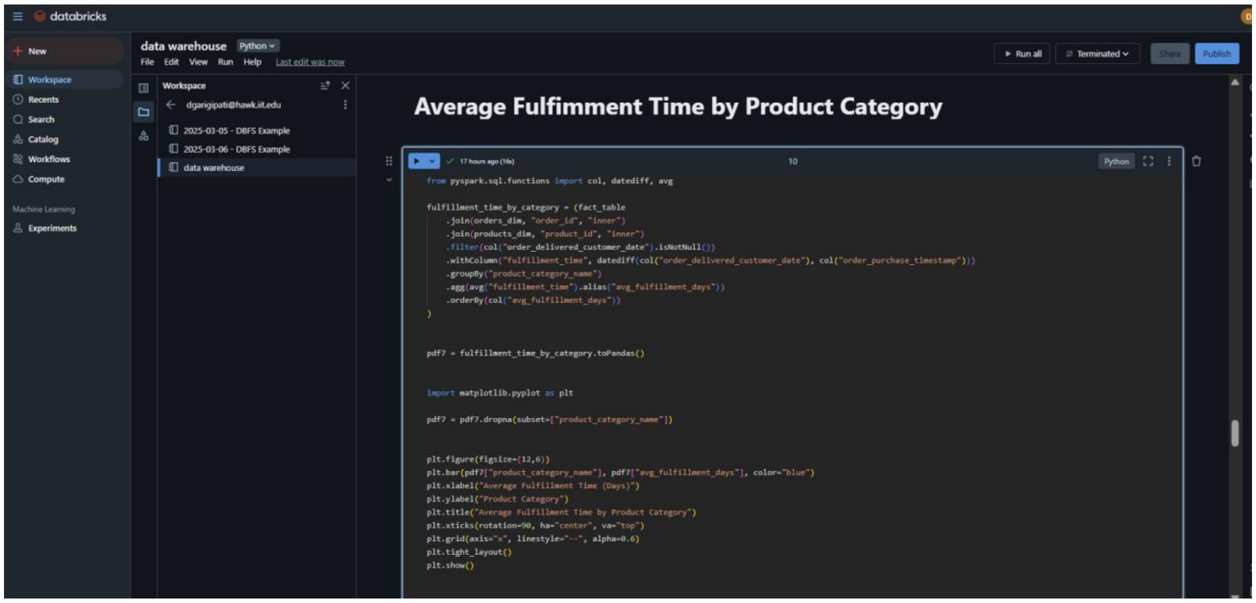
The **customer_dim** stores customer details, while **sellers_dim** contains seller information. **products_dim** holds product attributes, and its relationship with **prod_translation_dim** allows category name translations. **orders_dim** tracks various order-related timestamps, enabling order lifecycle analysis. **date_dim** facilitates time-based reporting, and **geolocation_dim** maps geographic details using zip code prefixes. The structure optimizes analytical queries by reducing data redundancy and improving performance.

DATA WAREHOUSE IMPLEMENTATION PROJECT

Analytical Queries:

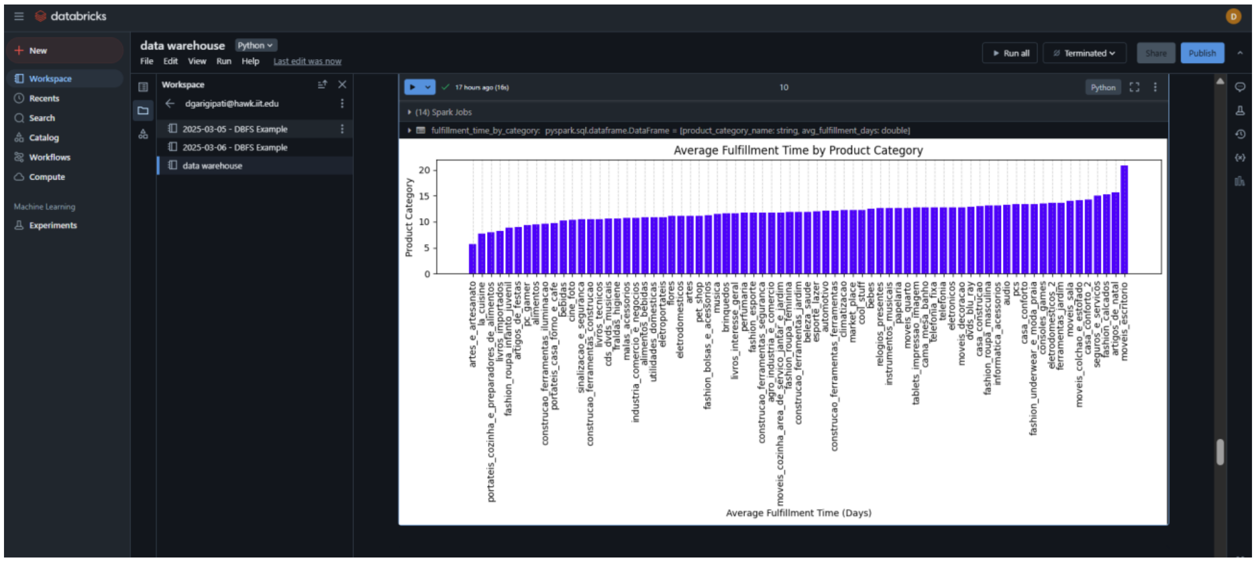
Average Fulfillment Time by Product Category

From the plot we can see that the artes_e_artesanato category have the lowest fulfillment within 5 days and then movies_escritorio have the highest fulfillment may take up to 20 days. With average fulfillment in 10 days range for all other categories.



Average Fulfimment Time by Product Category

```
from pyspark.sql.functions import col, datediff, avg

fulfillment_time_by_category = (fact_table
    .join(orders_dim, "order_id", "inner")
    .join(products_dim, "product_id", "inner")
    .filter(col("order_delivered_customer_date").isNotNull())
    .withColumn("fulfillment_time", datediff(col("order_delivered_customer_date"), col("order_purchase_timestamp")))
    .groupBy("product_category_name")
    .agg(avg("fulfillment_time").alias("avg_fulfillment_days"))
    .orderBy(col("avg_fulfillment_days"))
)


pdf7 = fulfillment_time_by_category.toPandas()


import matplotlib.pyplot as plt

pdf7 = pdf7.dropna(subset=["product_category_name"])


plt.figure(figsize=(12,6))
plt.bar(pdf7["product_category_name"], pdf7["avg_fulfillment_days"], color="blue")
plt.xlabel("Average Fulfillment Time (Days)")
plt.ylabel("Product Category")
plt.title("Average Fulfillment Time by Product Category")
plt.xticks(rotation=90, ha="center", va="top")
plt.grid(axis="x", linestyle="--", alpha=0.6)
plt.tight_layout()
plt.show()
```

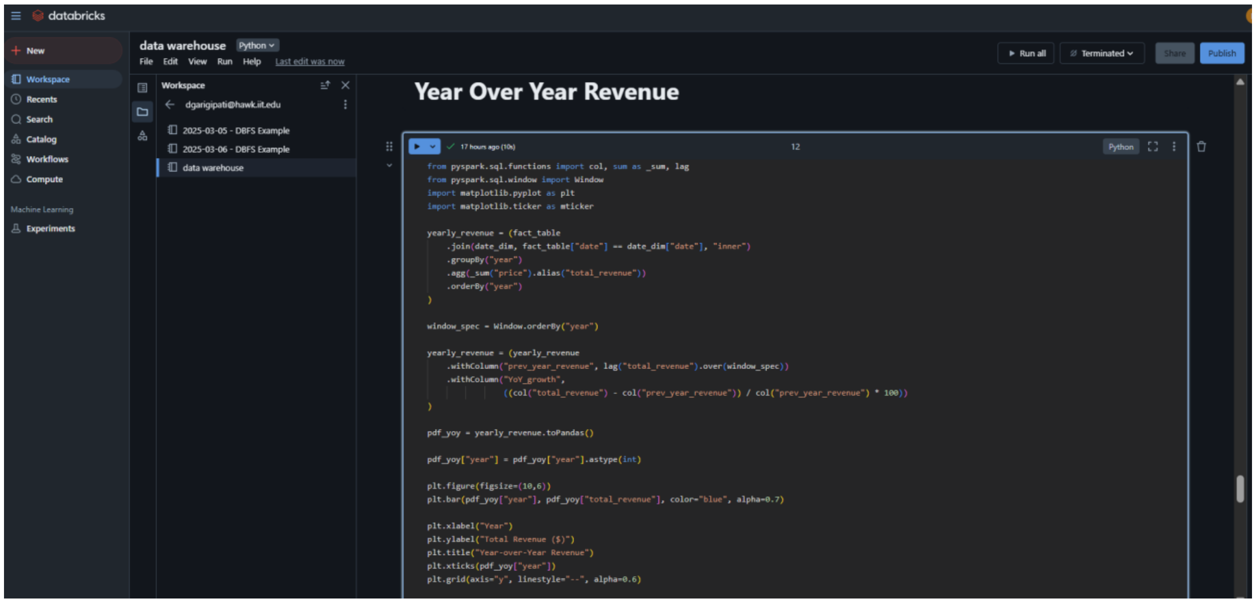


```
(14) Spark Jobs
fulfillment_time_by_category: pyspark.sql.dataframe.DataFrame = [product_category_name: string, avg_fulfillment_days: double]
```

DATA WAREHOUSE IMPLEMENTATION PROJECT

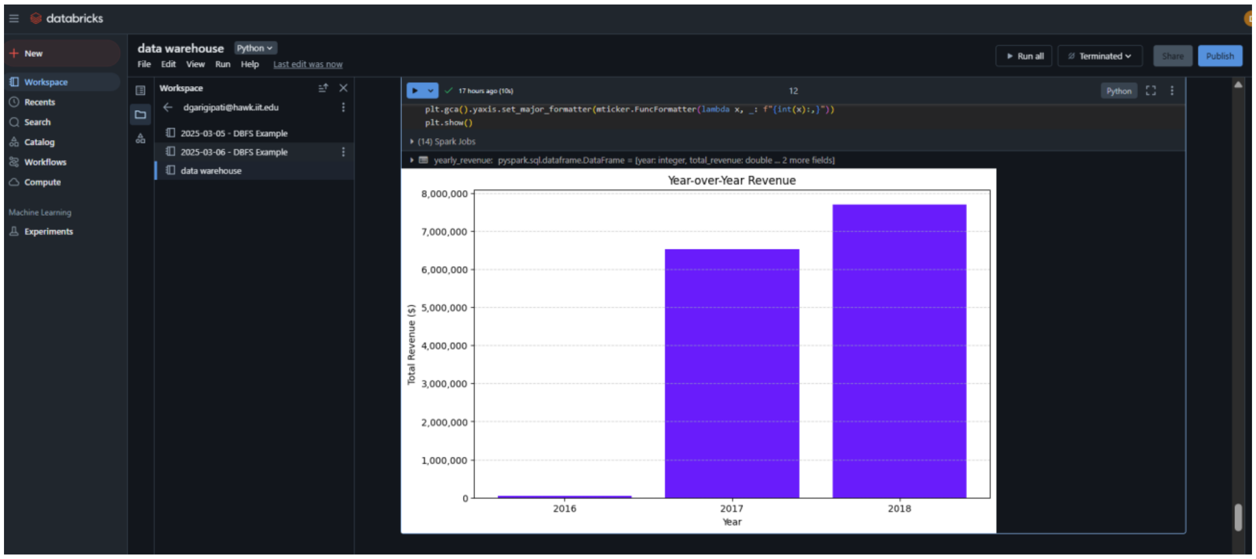
Year-Over-Year Revenue:

From the below graph we can see that there is huge gap between 2016 and 2017 this might be due to the fact that we are missing data form 2016. From 2017 to 2018 the revenue has increased from around 6500000 to about 8750000 that's equal to 30% increase.



Year Over Year Revenue

```python
from pyspark.sql.functions import col, sum as _sum, lag
from pyspark.sql.window import Window
import matplotlib.pyplot as plt
import matplotlib.ticker as mticker

yearly_revenue = (fact_table
    .join(date_dim, fact_table["date"] == date_dim["date"], "inner")
    .groupBy("year")
    .agg(_sum("price").alias("total_revenue"))
    .orderBy("year")
)

window_spec = Window.orderBy("year")

yearly_revenue = (yearly_revenue
    .withColumn("prev_year_revenue", lag("total_revenue").over(window_spec))
    .withColumn("YoY_growth",
                ((col("total_revenue") - col("prev_year_revenue")) / col("prev_year_revenue") * 100))
)

pdf_yoy = yearly_revenue.toPandas()

pdf_yoy["year"] = pdf_yoy["year"].astype(int)

plt.figure(figsize=(10,6))
plt.bar(pdf_yoy["year"], pdf_yoy["total_revenue"], color="blue", alpha=0.7)

plt.xlabel("Year")
plt.ylabel("Total Revenue ($)")
plt.title("Year-over-Year Revenue")
plt.xticks(pdf_yoy["year"])
plt.grid(axis="y", linestyle="--", alpha=0.6)
```



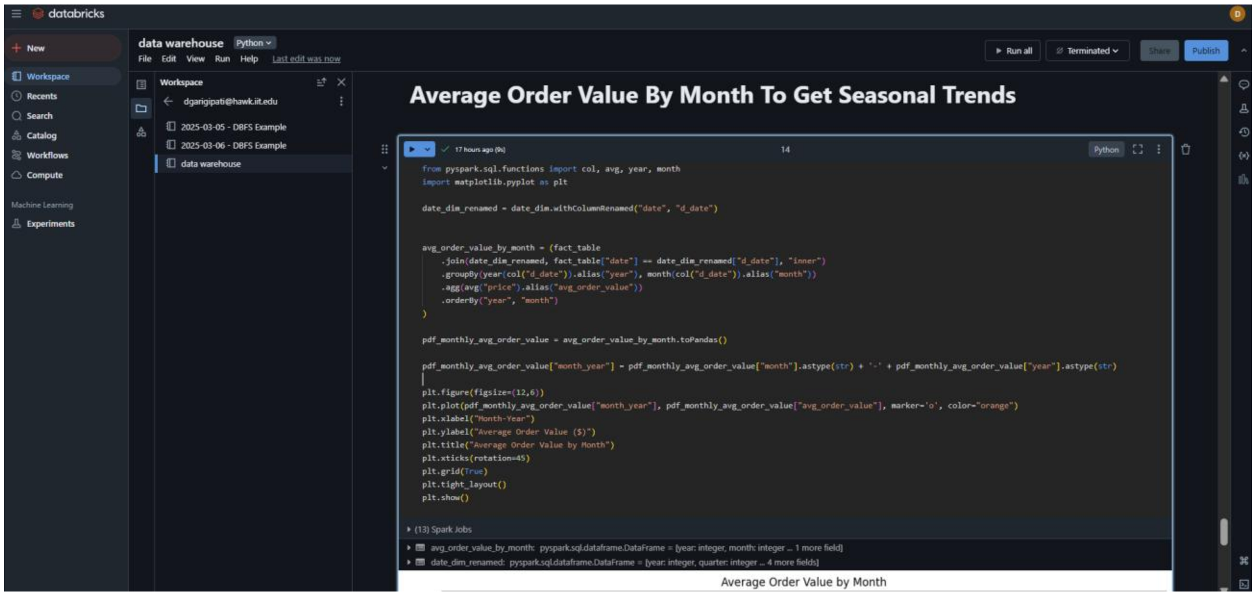
```python
plt.gca().yaxis.set_major_formatter(mticker.FuncFormatter(lambda x, _: f"{int(x):,}"))
plt.show()
```

```
(14) Spark Jobs
yearly_revenue:  pyspark.sql.dataframe.DataFrame = [year: integer, total_revenue: double ... 2 more fields]
```

DATA WAREHOUSE IMPLEMENTATION PROJECT

Average Order Value By Month To Get Seasonal Trends:

We can see that our data starts from 9/2016 this might've been contributed to the low revenue in previous analysis. We can see that from 1-2017 every month we have average orders of around 120/month.

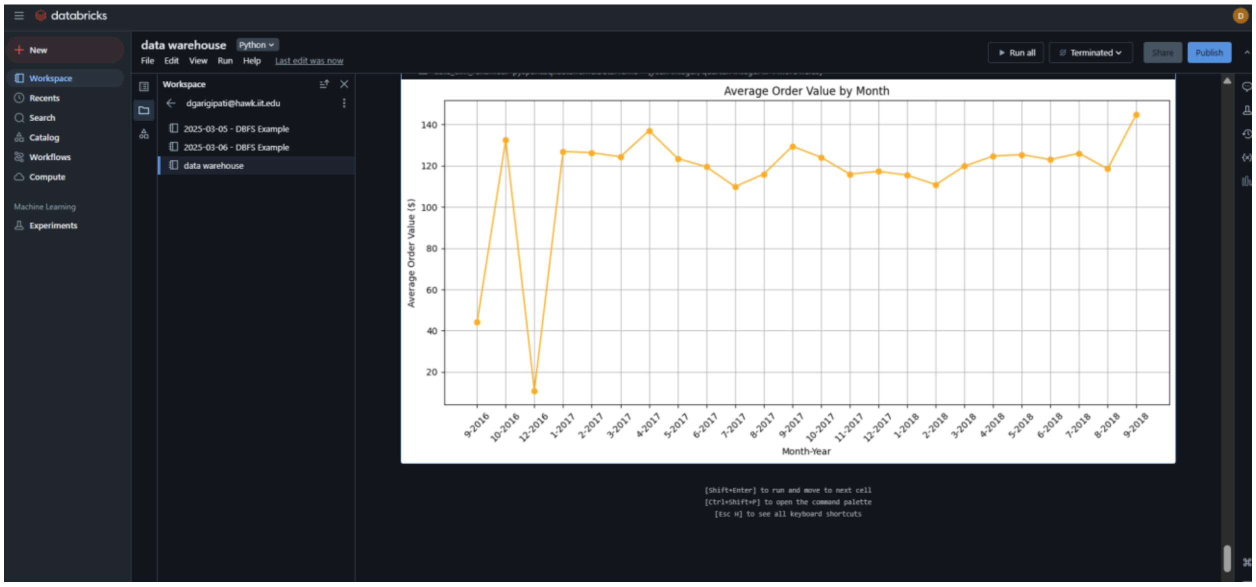


Average Order Value By Month To Get Seasonal Trends

```
from pyspark.sql.functions import col, avg, year, month
import matplotlib.pyplot as plt

date_dim_renamed = date_dim.withColumnRenamed("date", "d_date")


avg_order_value_by_month = (fact_table
    .join(date_dim_renamed, fact_table["date"] == date_dim_renamed["d_date"], "inner")
    .groupBy(year(col("d_date")).alias("year"), month(col("d_date")).alias("month"))
    .agg(avg("price").alias("avg_order_value"))
    .orderBy("year", "month")
)

pdf_monthly_avg_order_value = avg_order_value_by_month.toPandas()

pdf_monthly_avg_order_value["month_year"] = pdf_monthly_avg_order_value["month"].astype(str) + '-' + pdf_monthly_avg_order_value["year"].astype(str)

plt.figure(figsize=(12,6))
plt.plot(pdf_monthly_avg_order_value["month_year"], pdf_monthly_avg_order_value["avg_order_value"], marker='o', color="orange")
plt.xlabel("Month-Year")
plt.ylabel("Average Order Value ($)")
plt.title("Average Order Value by Month")
plt.xticks(rotation=45)
plt.grid(True)
plt.tight_layout()
plt.show()
```

```
▸ (13) Spark Jobs
▸ avg_order_value_by_month:  pyspark.sql.dataframe.DataFrame = [year: integer, month: integer ... 1 more field]
▸ date_dim_renamed:  pyspark.sql.dataframe.DataFrame = [year: integer, quarter: integer ... 4 more fields]
```

Average Order Value by Month



```
[Shift+Enter] to run and move to next cell
[Ctrl+Shift+P] to open the command palette
[Esc H] to see all keyboard shortcuts
```

DATA WAREHOUSE IMPLEMENTATION PROJECT

Summary:

The project implements a data warehouse by converting a raw, normalized e-commerce dataset into a star schema dimensional model. The central fact table (fact_sales) stores key transactional data, while surrounding dimension tables (e.g., customer_dim, products_dim, orders_dim) provide descriptive context for each dimension. Using PySpark, I've efficiently processed complex queries such as Year-over-Year revenue, top-selling categories, and fulfillment times. This approach streamlined the data model, improved performance, and enabled flexible business intelligence reporting.

Challenges included data quality issues, ambiguous column names, and data type mismatches, which were resolved by renaming columns and casting types. The star schema reduced query complexity and improved performance. Future improvements include implementing surrogate keys, enhancing the ETL pipeline for better data quality and optimizing query performance and report automation for better scalability and maintenance.