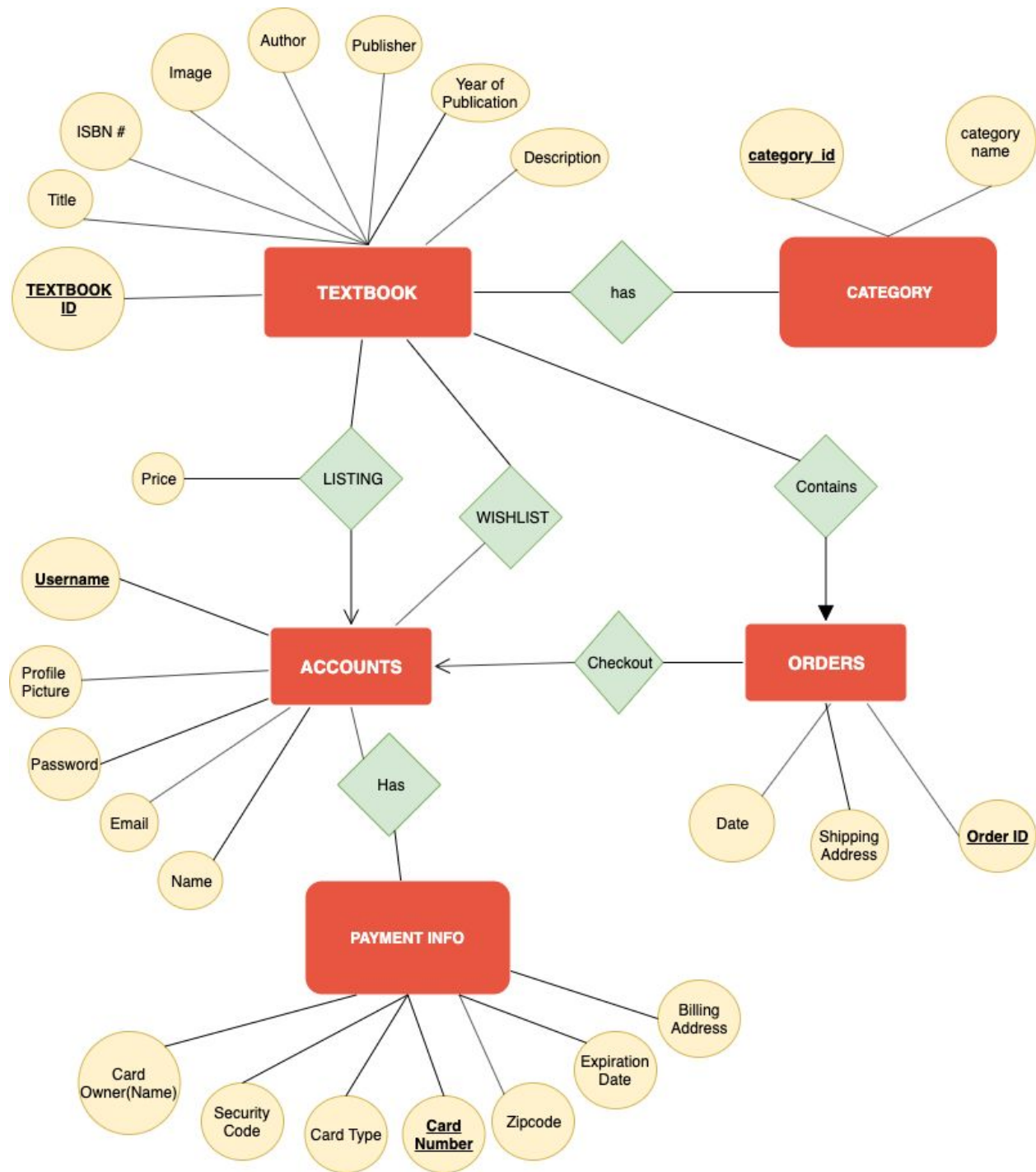


Project DB Model Design Revision

Team 25:
Au Tran
Benny Ooi
Aldrich Mangune



Entity Sets:

- 1) Textbook - Entity set representing a textbook. It contains attributes ISBN-13 #, Title, Image, Author, Publisher, Year of Publication, Description. It has an auto-increment primary key textbook_ID
- 2) Category - An entity set representing a category that a textbook belongs to. It has an auto-increment primary key category_id and attribute category name.
- 3) Orders - an entity set representing a transaction/order made (order histories). It contains auto-increment key Order ID, Date, and Shipping Address attribute.
- 4) Accounts - An entity set representing the account of the users of our application. It contains attributes Password, Email, Profile Picture, Name, and primary key Username
- 5) Payment Info is an entity set representing a credit/debit card users used to pay for transactions. It contains attributes Card Owner(Name), Security Code, Card Type, Zipcode, Expiration Date, Billing Address and primary key is the 16-digit Card Number.

Relationships:

- 1) Listing is a many-to-one relationship between Accounts and Textbooks. Each user would be able to add many textbook listings for sale and each textbook can only points to 1 account.
- 2) Wishlist is a many-to-many relationship between Accounts and Textbooks. Each user would be able to add many textbooks to their wishlist and each textbooks could be in many users' wishlist.
- 3) Checkout is a many-to-one relationship between Accounts and Order. The user would be able to checkout many orders while each order must be bind to 1 account.
- 4) Between Orders and Textbooks, there is a many-to-one relationship Contains where an order can contain multiple textbooks while each textbook can only belong to 1 order.
- 5) Between Textbook and Category, the many-to-many relationship Has represents that 1 category can have many textbooks and many textbooks can belong to many different categories.
- 6) Between Accounts and PaymentInfo, the many-to-many relationship Has represents that 1 account can have multiple payment methods and 1 credit card can be used for transactions in multiple accounts.

SCHEMA

We are using Django as part of our 3 tier architecture and unfortunately Django comes with an ORM that implements a database table as an object in Python. This mean that I have to specify my schemas as classes in Python and Django will automatically convert the classes into MySQL tables and each object instance I created will be converted to a new table instance in MySQL database.

- 1.) The problem is that Django does not support the use of composite primary key. It requires that only 1 column can be primary key maximum. Therefore, in the tables that represent relationship Django automatically add a field id as the auto-increment primary key. For example, relationship Wishlist is a relationship that connects entity set Account and Textbook. The table for Wishlist will become Wishlist(id, Account_key, Textbook_key). However, Django lets me add a constraint specifying that no instances of Wishlist can have the same Account_key and Textbook_key so it behave the same as a primary key. The only setback is that those relationship will have a mandatory auto-increment primary key id, the behavior will be the same.
- 2.) Django automates the conversion of my schemas into Database tables. However, the naming convention can be a little disambiguous. For example, Wishlist again is a relationship connecting entity set Account and Textbook so the table will contain the key of these 2 attributes. Account have primary key Username and Textbook have primary key Textbook ID. When Django convert this schema into a table, it will create a table Wishlist with column names Account_id and Textbook_id. This Account_id does actually contain the primary key Username so I hope it doesn't confuse you thinking its some other attribute.
- 3.) Some of the table have an Image attribute that I didn't add to the table.

SCHEMAS

store

Tables

Account

Account_Has_Payme...

auth_group

auth_group_permissi...

auth_permission

auth_user

auth_user_groups

auth_user_user_permi...

Category

Category_Has_Textbo...

Checkout

django_admin_log

django_content_type

django_migrations

Object Info

Session

Table: Account

Columns:

<u>Username</u>	varchar(20) PK
Profile_Picture	varchar(100)
Password	varchar(20)
Email	varchar(50)
Name	varchar(25)

1

SELECT * FROM store.Account;

Limit to 1000 rows

100%

1:1

Result Grid

Filter Rows:

Edit:

Export/Import:

Username	Profile_Picture	Password	Email	Name
angela44		wP+F2Pmv5(josephgreene@hotmail.com	Robert Payne
brettosborn		NY5M8k2n)N	tmartin@mcdownell-reynolds.com	Elizabeth Matthews
briggsdouglas		ytX0i5Xcm(wwashington@yahoo.com	Steven Jensen
christopherlee		M6+MO#dg&#	osmith@campbell.com	Lauren Russell
deborahfrench		D6UczV+L@&	kristinpruitt@jenkins.com	Tonya Edwards
jeanettelewis		IH7M#+Fuu_	jeremy22@carter.com	James Harvey
jljong		+8f5l+ygml	dyerrichard@salazar-moss.com	Stephanie Brown
lindsay10		%xDJoKVbh5	yrichardson@levine-hopkins.org	David Brown
luismoree		#u2KzPkxu4	gjohnson@vaughn.com	Wayne Glass
mariebailey		'8V&00m+(e	hessjessica@yahoo.com	Kim Gillespie
marisa54		+4WYga3o57	julie84@gmail.com	Maria Hull
michaelperez		9NiTks7#y	phillipallen@yahoo.com	Wanda Cox
plittie		&8Y3VCih4M	zcaldwell@gmail.com	Jennifer James
ramosjorge		%08pO89ftz	fcrosby@gmail.com	Eric Barnes
timothy19		#3C1(bbW2o	nray@gmail.com	Michael Gibson
	HULL	HULL	HULL	HULL

Account 1

Apply

Accounts(**Username**, Profile Picture, Password, Email, Name)

SCHEMAS

Filter objects

- django_admin_log
- django_content_type
- django_migrations
- django_session
- Listing
- Order
- Order_Contain_Textb...
- PaymentInfo
- Textbook**
- Wishlist
- Views
- Stored Procedures
- Functions
- sys
- test

Object Info **Session**

Table: Textbook

Columns:

Textbook_ID	int(11) AI PK
ISBN	bigint(20)
Title	varchar(100)
Image	varchar(100)
Author	varchar(30)
Publisher	varchar(20)
Year_Of_Publicatio n	int(11)

Query Completed

Limit to 1000 rows

1 • `SELECT * FROM store.Textbook;`

100% 1:1

Result Grid Filter Rows: Search Edit: Export/Import:

Textbook_ID	ISBN	Title	Image	Author	Publisher	Year_Of_Publication	Description
1	9781234567897	Modern Dental Assisting		Debbie S. Robinson	Generic	2000	example
2	9781281684462	Pilates Anatomy		Rael Isacowitz	Generic	2016	example
3	9781596553095	The Hubbles Cosmos		David Devorkin	Generic	2001	example
4	9780415517850	Mechanical Engineering		John Bird	Generic	2007	example
5	9780984782857	Cracking the Coding Interview		Gayle McDowell	CheerCup	2005	example
6	9781426220586	National Geographic Atlas of the World		Alex Tait	Generic	1998	example
7	9781260125412	Geography of Louisiana		Elaine Yodis	McGraw-Hill	2016	example
8	9780321909107	Conceptual Physics		Paul G. Hewitt	Generic	2004	example
9	9780321743268	Human Anatomy and Physiology		Elaine Marieb	Generic	2010	example
10	9781285057095	Larson Calculus		Ron Larson	Generic	2006	example
11	9781464111723	Exploring Psychology		David G. Myers	Generic	2011	example
12	9780815344322	Molecular Biology of the Cell		Bruce Alberts	Generic	2012	example
13	9781133311294	Steps to Writing Well		Jean Wyrick	Generic	2003	example
14	9780312679484	Technical Communication		Mike Markel	Generic	2007	example
15	9780312601430	A Writer's Reference		Diana Hacker	Generic	2008	example
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Textbook 1 Apply

Action Output

	Time	Action	Response	Duration / F
1	10:52:51	SELECT * FROM store.Textbook LIMIT 0, 1000	15 row(s) returned	0.00036 sec

Textbook(**Textbook_ID**, ISBN, Title, Image, Author, Publisher, Year_Of_Publication, Description)

SCHEMAS

Filter objects

- Checkout
- django_admin_log
- django_content_type
- django_migrations
- django_session
- Listing
- Order**
 - Order_Contain_Textb...
 - PaymentInfo
 - Textbook
 - Wishlist
 - Views
 - Stored Procedures
 - Functions
- sys
- test

Object Info **Session**

Table: Order

Columns:

Order_ID	int(11) AI PK
Date	datetime(6)
Shipping_Address	varchar(100)

1 • `SELECT * FROM store.Order;`

100% 1:1

Result Grid Filter Rows: Search Edit: Export/Import:

Order_ID	Date	Shipping_Address
1	2019-10-15 05:31:42.000000	3670 Ramirez Extension Suite 622
2	2019-10-15 05:31:42.000000	3670 Ramirez Extension Suite 622
3	2019-10-15 05:32:07.000000	4441 Debbie Station Angelaland, ID 25464
4	2019-10-15 05:32:19.000000	830 Tom Dale Suite 313 Nathanielberg, MS 36935
5	2019-10-15 05:32:32.000000	59039 Rocha Trace Smithfort, IL 76188
6	2019-10-15 05:32:42.000000	258 Barker Skyway Navarropport, OH 80059
7	2019-10-15 05:32:52.000000	63314 Miller Well Port Travishaven, CO 18530
8	2019-10-15 05:33:16.000000	64449 Price Tunnel Jeffreyfort, KS 29371
9	2019-10-15 05:33:21.000000	9565 Andrew Drive Apt. 118 Port Chelsea, NV 4...
10	2019-10-15 05:33:34.000000	110 Booth Rapids Lake Alexanderland, FL 40297
11	2019-10-15 05:33:46.000000	44082 Margaret Dam Andreabury, HI 02464
12	2019-10-15 05:34:02.000000	614 Jesse Junctions Campbellshire, TX 03242
13	2019-10-15 05:34:11.000000	7712 Griffin Knoll Port Joannaton, OH 87041
14	2019-10-15 05:34:23.000000	137 Wayne Vista Andersonburgh, CT 26095
15	2019-10-15 05:34:39.000000	3295 Bailey Overpass North Kenneth, SC 88657
NULL	NULL	NULL

Order 1 Apply R

Query Completed

Order(Order_ID, Date, Shipping_Address)

SCHEMAS

Filter objects

- Checkout
- django_admin_log
- django_content_type
- django_migrations
- django_session
- Listing
- Order
- Order_Contain_Textb...
- PaymentInfo
- Textbook
- Wishlist
- Views
- Stored Procedures
- Functions
- sys
- test

Object Info **Session**

Table: PaymentInfo

Columns:

- Card_Number: bigint(20) PK
- Card_Name: varchar(30)
- Security_Code: int(11)
- Card_Type: varchar(50)
- Zip_code: int(11)
- Expiration_Date: varchar(10)
- Billing_Address: varchar(100)

1 • `SELECT * FROM store.PaymentInfo;`

100% 1:1

Result Grid Filter Rows: Search Edit: Export/Import:

Card_Number	Card_Name	Security_Code	Card_Type	Zip_code	Expiration_Date	Billing_Address
38600749532993	Andre Brown	884	VISA 19 digit	59135	11/27	27239 Riddle Courts Apt. 937 Lake Sara, NJ 08600
5179014261159680	Miguel Davies	639	VISA 19 digit	95523	05/29	3551 Jackson Lock North Briannaland, NH 99442
3503010966696424	Roger Hopkins	1834	JCB 15 digit	83337	04/25	377 Debbie Terrace Suite 451 West Monica, GA 88166
3579537579034584	Russell Ortiz	633	VISA 16 digit	69896	04/22	4344 Jennifer Square South Jessica, ND 62478
341242618262352	Andrew Miller	356	VISA 16 digit	24667	04/27	56468 Barnett Locks Jessicatown, OH 67228
4684868848667973	Christopher Smith	814	JCB 15 digit	52132	03/21	570 Wallace Corners Apt. 863 West Anthonyland, CT 96228
372846958221588	Pamela Greene	18	VISA 19 digit	65268	10/28	6056 Jean Flats South Martinhaven, OK 08466
4986402237129182	Mrs. Ann Knight MD	923	JCB 16 digit	872	04/27	651 Lisa Overpass South Paul, WY 43647
4021548591247551	Charles Hayden	300	VISA 16 digit	50284	11/22	7091 Jason Stream Suite 943 New Ronald, MI 20593
3523463313719033	Christopher Hayden	598	JCB 16 digit	80772	08/29	80943 Erin Fall Nelsonton, AZ 46448
5581604894286686	Stephanie Townsend	356	Discover	90897	08/28	820 Allen Views Russellport, NJ 02708
5330323804121065	Jennifer Jackson	188	American E...	28845	01/21	PSC 9833, Box 3909 APO AP 33398
601112855711806	Robert Pacheco	308	Maestro	19155	06/24	USNS Walsh FPO AE 44647
452731112160854	Brenda Cobb	471	VISA 16 digit	99151	06/29	USNV Medina FPO AE 37519
340057068170736	David Barry	983	Mastercard	64820	03/22	USS Brown FPO AP 31512
NULL	NULL	NULL	NULL	NULL	NULL	NULL

PaymentInfo 1

Action Output

Time	Action	Response	Duration / Fe
15 10:32:39	SELECT * FROM store.Order LIMIT 0, 1000	15 row(s) returned	0.0097 sec /

Query Completed

PaymentInfo(**Card Number**, Card Name, Security Code, Card Type, Zipcode, Expiration Date, Billing Address)

SCHEMAS

Filter objects

- store
 - Tables
 - Account
 - Account_Has_Payme...
 - auth_group
 - auth_group_permi...
 - auth_permission
 - auth_user
 - auth_user_groups
 - auth_user_user_permi...
 - Category
 - Category_Has_Textbo...
 - Checkout
 - django_admin_log
 - django_content_type
 - django_migrations

Object Info **Session**

Table: Account_Has_PaymentInfo

Columns:

- id: int(11) AI PK
- Account_id: varchar(20)
- PaymentInfo_id: bigint(20)

1 • `SELECT * FROM store.Account_Has_PaymentInfo;`

100% 1:1

Result Grid Filter Rows: Search Edit: Export/Import:

id	Account_id	PaymentInfo_id
7	angela44	38600749532993
6	brettosborn	5581604894286686
2	briggsdouglas	38600749532993
1	briggsdouglas	3503010966696424
3	briggsdouglas	4986402237129182
14	christopherlee	341242618262352
13	jeanettelewis	372846958221588
12	jiong	5581604894286686
9	mariebailey	3503010966696424
11	mariebailey	4021548591247551
4	marisa54	372846958221588
15	michaelperez	3579537579034584
5	michaelperez	4684868848667973
10	plittle	601112855711806
8	ramosjorge	3579537579034584
NULL	NULL	NULL

Account_Has_PaymentInfo 1

Action Output

Time	Action	Response	Duration / Fe
1 10:59:22	SELECT * FROM store.Account_Has_PaymentInfo LIMIT 0, 1000	15 row(s) returned	0.00079 sec

Query Completed

Account_Has_PaymentInfo(**id** Account_id, Paymentinfo_id)

The auto increment id primary key is required and automatically added by Django.

This Account_id is primary key of Account which is Username, and Paymentinfo_id is actually primary key of PaymentInfo which is Card Number.

SCHEMAS

Filter objects

- Account
- Account_Has_Payme...
- auth_group
- auth_group_permissi...
- auth_permission
- auth_user
- auth_user_groups
- auth_user_user_permi...
- Category
- Category_Has_Textbo...
- Checkout
- django_admin_log
- django_content_type
- django_migrations
- django_session
- listing

Object Info

Session

Table: Category_Has_Textbook

Columns:

id

int(11) AI

PK

Category_id

int(11)

Textbook_id

int(11)

100%

1:1

Result Grid

Filter Rows:

Search

Edit:

Export/Import:

id	Category_id	Textbook_id
12	2	12
1	3	5
2	3	6
9	3	12
5	5	12
15	7	3
7	10	5
4	10	10
14	11	1
13	12	5
8	12	14
11	13	5
3	13	6
10	14	3
6	15	4
NULL	NULL	NULL

Category_Has_Textbook 1

Apply

Query Completed

Category_Has_Textbook(id, Category_id, Textbook_id)

The auto increment id primary key is required and automatically added by Django.

[illegible]

SCHEMAS

Filter objects

- Checkout
- django_admin_log
- django_content_type
- django_migrations
- django_session
- Listing
- Order
- Order_Contain_Textb...
- PaymentInfo
- Textbook
- Wishlist**
- Views
- Stored Procedures
- Functions
- sys
- test

Object Info **Session**

Table: Wishlist

Columns:

- id** int(11) AI PK
- Account_id** varchar(20)
- Textbook_id** int(11)

Limit to 1000 rows

1 • `SELECT * FROM store.Wishlist;`

100% 1:1

Result Grid Filter Rows: Search Edit: Export/Import:

id	Account_id	Textbook_id
1	angela44	7
2	deborahfrench	10
10	jeanettelewis	5
9	jeanettelewis	12
16	jeanettelewis	15
15	jlong	12
3	mariebailey	3
5	marisa54	6
14	marisa54	13
11	marisa54	14
12	marisa54	15
7	ramosjorge	2
8	ramosjorge	3
6	ramosjorge	4
4	ramosjorge	5
13	timothy19	15
NULL	NULL	NULL

Wishlist 1 Apply

Action Output

	Time	Action	Response	Duration / F
20	10:35:49	SELECT * FROM store.Wishlist LIMIT 0, 1000	16 row(s) returned	0.00067 sec

Query Completed

Wishlist(**id**, Account_id, Textbook_id)

The auto increment id primary key is required and automatically added by Django.

This Account_id is primary key of Account which is Username

SCHEMAS

Filter objects

- Account
- Account_Has_Payme...
- auth_group
- auth_group_permissi...
- auth_permission
- auth_user
- auth_user_groups
- auth_user_user_permi...
- Category
- Category_Has_Textbo...
- Checkout
- django_admin_log
- django_content_type
- django_migrations
- django_session
- Listing

Object Info Session

Table: **Checkout**

Columns:

- id** int(11) AI PK
- Account_id** varchar(20)
- Order_id** int(11)

1 • **SELECT * FROM store.Checkout;**

Limit to 1000 rows

100% 1:1

Result Grid Filter Rows: Search Edit: Export/Import:

id	Account_id	Order_id
6	angela44	1
5	briggsdouglas	2
4	briggsdouglas	3
3	christopherlee	4
11	christopherlee	11
2	deborahfrench	5
1	deborahfrench	6
9	jeanettelewis	9
7	luismoore	7
14	luismoore	14
8	mariebailey	8
15	mariebailey	15
10	michaelperez	10
12	ramosjorge	12
13	timothy19	13
NULL	NULL	NULL

Checkout 1 Apply

Query Completed

Checkout(**id**, Account_id, Order_id)

The auto increment id primary key is required and automatically added by Django.

This Account_id is primary key of Account which is Username.

SCHEMAS

Filter objects

- Checkout
- django_admin_log
- django_content_type
- django_migrations
- django_session
- Listing
- Order
- Order_Contain_Textb...
- PaymentInfo
- Textbook
- Wishlist
- Views
- Stored Procedures
- Functions
- sys
- test

Object Info Session

Table: **Order_Contain_Textbook**

Columns:

- id** int(11) AI PK
- Order_id** int(11)
- Textbook_id** int(11)

1 • `SELECT * FROM store.Order_Contain_Textbook;`

100% 1:1

Result Grid Filter Rows: Search Edit: Export/Import:

id	Order_id	Textbook_id
1	1	1
2	2	2
3	3	3
4	1	4
5	10	5
6	11	6
7	6	7
8	13	8
9	14	9
10	15	10
11	14	11
12	14	12
13	3	13
14	1	14
15	6	15
NULL	NULL	NULL

Order_Contain_Textbook 1 Apply

Query Completed

Order_Contain_Textbook(id, Order_id, Textbook_id)

The auto increment id primary key is required and automatically added by Django.

SCHEMAS

Filter objects

- Account
- Account_Has_Payme...
- auth_group
- auth_group_permi...
- auth_permission
- auth_user
- auth_user_groups
- auth_user_user_permi...
- Category**
- Category_Has_Textbo...
- Checkout
- django_admin_log
- django_content_type
- django_migrations
- django_session
- Listing

Object Info Session

Table: **Category**

Columns:

- Category_ID int(11) AI PK
- Category_Name varchar(20)

1 • `SELECT * FROM store.Category;`

100% 1:1

Result Grid Filter Rows: Search Edit: Export/Import:

Category_ID	Category_Name
1	Science
2	Arts
3	Computers & Tech
4	Cooking
5	Hobbies & Crafts
6	Education
7	Horror
8	Mathematics
9	Horror
10	Mysteries
11	Health & Fitness
12	Social Sciences
13	Law
14	Medicine
15	Engineering
NULL	NULL

Category 1 Apply R

Query Completed

Category(Category_ID, Category_Name)