Sean Kennedy: Database Design HW 4

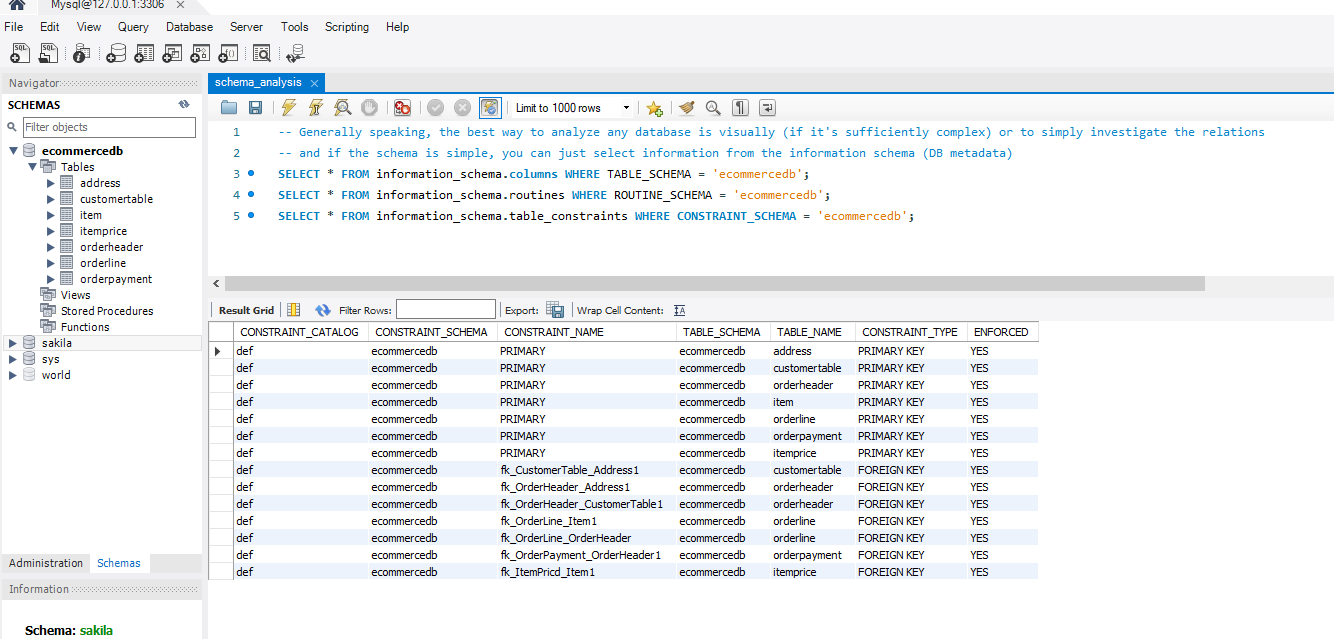
## Setup/Create/Inspect DB:

Generally speaking, the best way to analyze any database is visually (if it's sufficiently complex) or to simply investigate the relations and if the schema is simple, you can just select information from the information schema (DB metadata)

SELECT \* FROM information\_schema.columns WHERE TABLE\_SCHEMA = 'ecommercedb';

SELECT \* FROM information\_schema.routines WHERE ROUTINE\_SCHEMA = 'ecommercedb';

SELECT \* FROM information\_schema.table\_constraints WHERE CONSTRAINT\_SCHEMA = 'ecommercedb';

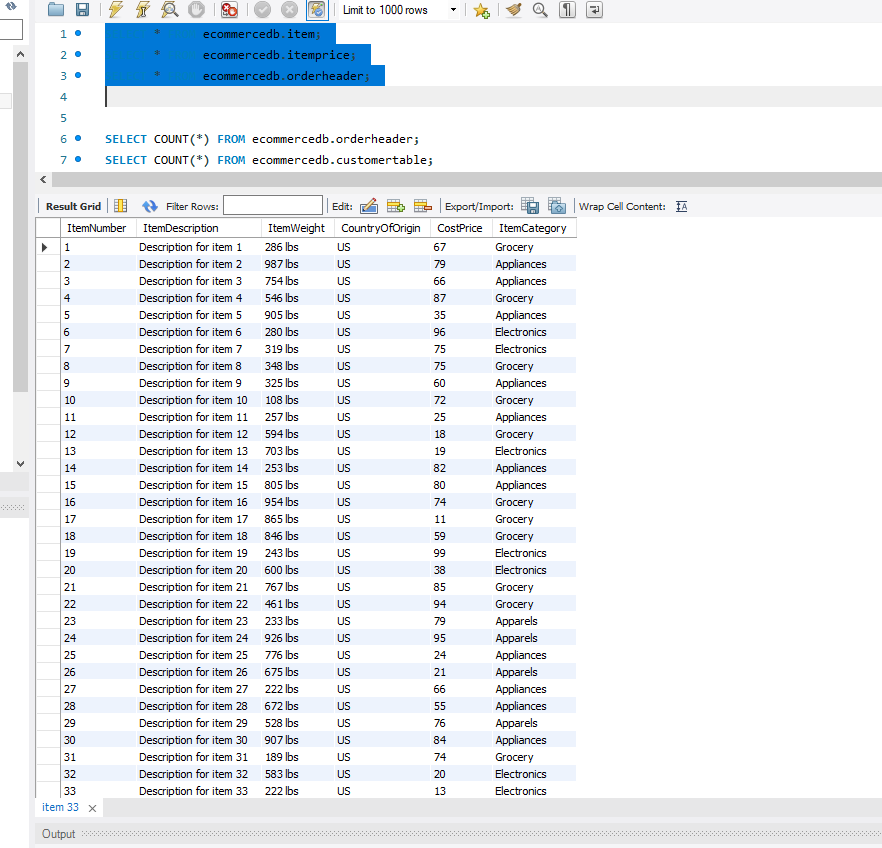


## Load Data:

SELECT \* FROM ecommercedb.item;

SELECT \* FROM ecommercedb.itemprice;

SELECT \* FROM ecommercedb.orderheader;



## Exercises:

SELECT COUNT(\*) FROM ecommercedb.orderheader;

SELECT COUNT(\*) FROM ecommercedb.customertable;

-- total number of customers that have placed an order;

SELECT COUNT(\*) FROM ecommercedb.customertable WHERE CustomerID

IN (SELECT CustomerTable\_CustomerID FROM ecommercedb.orderheader);

-- unique zipcodes

SELECT DISTINCT Zip FROM ecommercedb.address;

-- ten most frequent items

SELECT \* FROM (

SELECT Item\_ItemNumber, COUNT(\*) FROM orderline GROUP BY Item\_ItemNumber

ORDER BY COUNT(\*) DESC) as temp LIMIT 10;

## Design Question:

### How would you change your database design?

* First things first – you need to have your end users specify the following:
  + The recipient’s contact details
  + The recipient’s address
* Once they are in the database – I would simply add two new columns to the **orderline** table:
  + OrderLine\_CustomerId (fk to customer, nullable)
  + OrderLine\_AddressId (fk to Address, nullable)
* At which point I would use views to access current information about the states of each order – see **vw\_orders –** most of the other business logic would be implemented via coalesce statements where necessary (i.e shipping etc.)