Dr Simon D'Alfonso

INFO 90002 Database Systems & Information Modelling

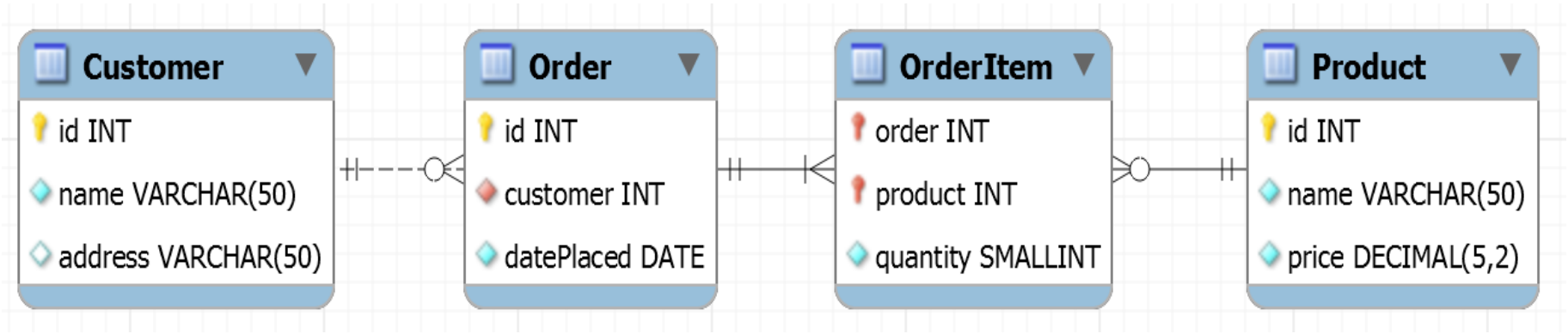
Week 04

Data Dictionaries

What is a data dictionary

- A data dictionary is how a RDMS maintains data about databases, their tables and their relationships. Basically, it is meta-data about the database structure.
- Data Dictionary consists of the following typical information:
  - Names of the tables in the database
  - Names, type and other information on the columns for each table
  - Constraints of a table. Keys, Relationships, etc.
  - Owner and authorised users of the table
  - Last accessed information of objects
  - Last updated information of objects
  - Engines, character types

Example basic dictionary structure



- Recall our orders example

Basic data dictionary

- For each table in database, provide information on its structure (column names, types, properties), indexes and foreign keys. This is the type of information that is found in MySQL Workbench
- Structure:

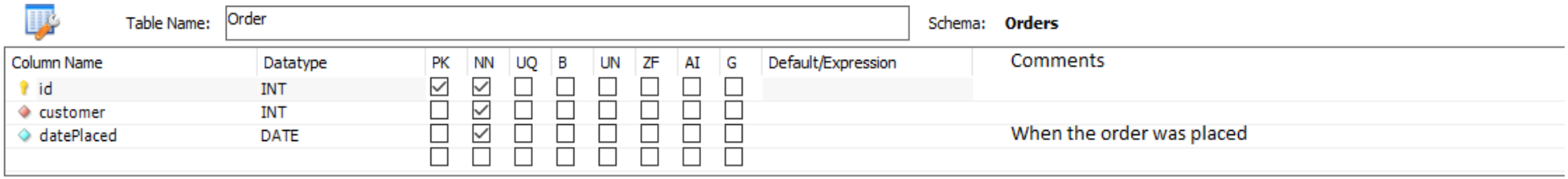


Table Name: Order Schema: **Orders**

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression	Comments
id	INT	☑	☑	☐	☐	☐	☐	☐	☐		
customer	INT	☐	☑	☐	☐	☐	☐	☐	☐		
datePlaced	DATE	☐	☑	☐	☐	☐	☐	☐	☐		When the order was placed
		☐	☐	☐	☐	☐	☐	☐	☐		

- Indexes:

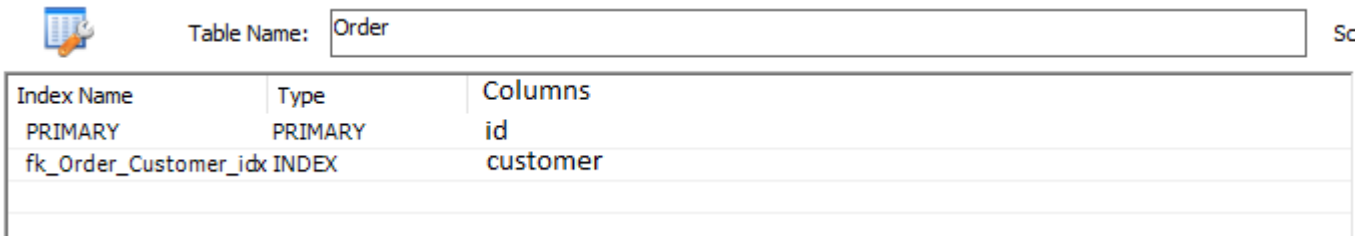


Table Name: Order Sc

Index Name	Type	Columns
PRIMARY	PRIMARY	id
fk_Order_Customer_idx	INDEX	customer

Basic data dictionary

- Foreign Keys

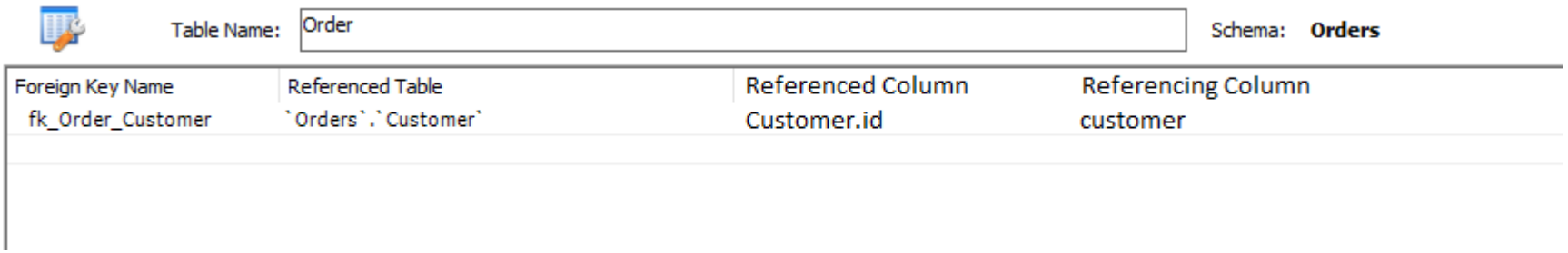


Table Name: Order

Schema: **Orders**

Foreign Key Name	Referenced Table	Referenced Column	Referencing Column
fk_Order_Customer	`Orders`.`Customer`	Customer.id	customer

Data dictionary structure

- How is a data dictionary stored?
- The main way is as a schema and tables within the DBMS itself.
- Though there are other ways, such as in metadata files external to the DBMS.
- MySQL Server incorporates a transactional data dictionary that stores information about database objects. In previous MySQL releases, dictionary data was stored in metadata files, non-transactional tables, and storage engine-specific data dictionaries.

MySQL Data Dictionary

- The `mysql` system schema contains information required by the MySQL server as it runs. This database contains data dictionary tables and system tables.
- Data dictionary tables are protected and may only be accessed in debug builds of MySQL.
- Thus, data dictionary tables are invisible. They cannot be read with SELECT, do not appear in the output of SHOW TABLES and so forth.
- However, there is something called INFORMATION_SCHEMA views that provides corresponding dictionary information. Conceptually, the INFORMATION_SCHEMA provides a view through which MySQL exposes data dictionary metadata.

information_schema views

- A database view is like a virtual table, a searchable object defined by a query and that can itself be queried like an actual table.
- The information schema (information_schema) is an ANSI-standard set of read-only views which provide information about all of the tables, columns, constraints, etc in a database management system.
- Some main views:
  - COLUMNS – Return one row for each column the current user has access to use in the current database. This view can be used to determine the data type and table the column is defined for use in.
  - TABLES – Return one row for each table the users has access to use within the current database. Note, both tables and views are returned using the TABLES view.
  - The REFERENTIAL_CONSTRAINTS table provides information about foreign keys.
  - The KEY_COLUMN_USAGE table describes which key columns have constraints and information about those constraints.

information_schema examples

- As mentioned, `mysql` dictionary is not directly accessible

```
1  mysql> SELECT * FROM mysql.schemata;
2  ERROR 3554 (HY000): Access to data dictionary table 'mysql.schemata' is rejected.
```

- Instead, to browse tables/schemata, do “SELECT * FROM INFORMATION_SCHEMA.SCHEMATA”

CATALOG_NAME	SCHEMA_NAME	DEFAULT_CHARACTER_SET_	DEFAULT_COLLATION_NAME	SQL_PATH
def	information_schema	utf8	utf8_general_ci	NULL
def	aware_test	utf8	utf8_general_ci	NULL
def	bank	utf8	utf8_general_ci	NULL
def	mysql	utf8	utf8_general_ci	NULL
def	orders	utf8	utf8_general_ci	NULL
def	performance_schema	utf8	utf8_general_ci	NULL
def	socialmedia	utf8	utf8_general_ci	NULL
def	sys	utf8	utf8_general_ci	NULL

information_schema examples

- SELECT * FROM `TABLES` WHERE TABLE_SCHEMA = 'orders'

TABLE_CATALOG	TABLE_SCHEMA	TABLE_NAME	TABLE_TYPE	ENGINE	VERSION	ROW_FORMAT	TABLE_ROWS	AVG_I
def	orders	customer	BASE TABLE	InnoDB	10	Dynamic	2	8192
def	orders	order	BASE TABLE	InnoDB	10	Dynamic	5	3276
def	orders	orderitem	BASE TABLE	InnoDB	10	Dynamic	11	1489
def	orders	product	BASE TABLE	InnoDB	10	Dynamic	4	4096

- SELECT TABLE_NAME, COLUMN_NAME, CONSTRAINT_NAME, REFERENCED_TABLE_NAME, REFERENCED_COLUMN_NAME FROM INFORMATION_SCHEMA.KEY_COLUMN_USAGE WHERE CONSTRAINT_SCHEMA = 'orders';

TABLE_NAME	COLUMN_NAME	CONSTRAINT_NAME	REFERENCED_TABLE_NAME	REFERENCED_COLUMN_NAM
customer	id	PRIMARY	NULL	NULL
order	id	PRIMARY	NULL	NULL
order	customer	fk_Order_Customer	customer	id
orderitem	order	PRIMARY	NULL	NULL
orderitem	product	PRIMARY	NULL	NULL
orderitem	order	fk_OrderItem_Order1	order	id
orderitem	product	fk_OrderItem_Product1	product	id
product	id	PRIMARY	NULL	NULL

information_schema

- A use for information_schema?

Next Week

- Next week: normalisation and more SQL.