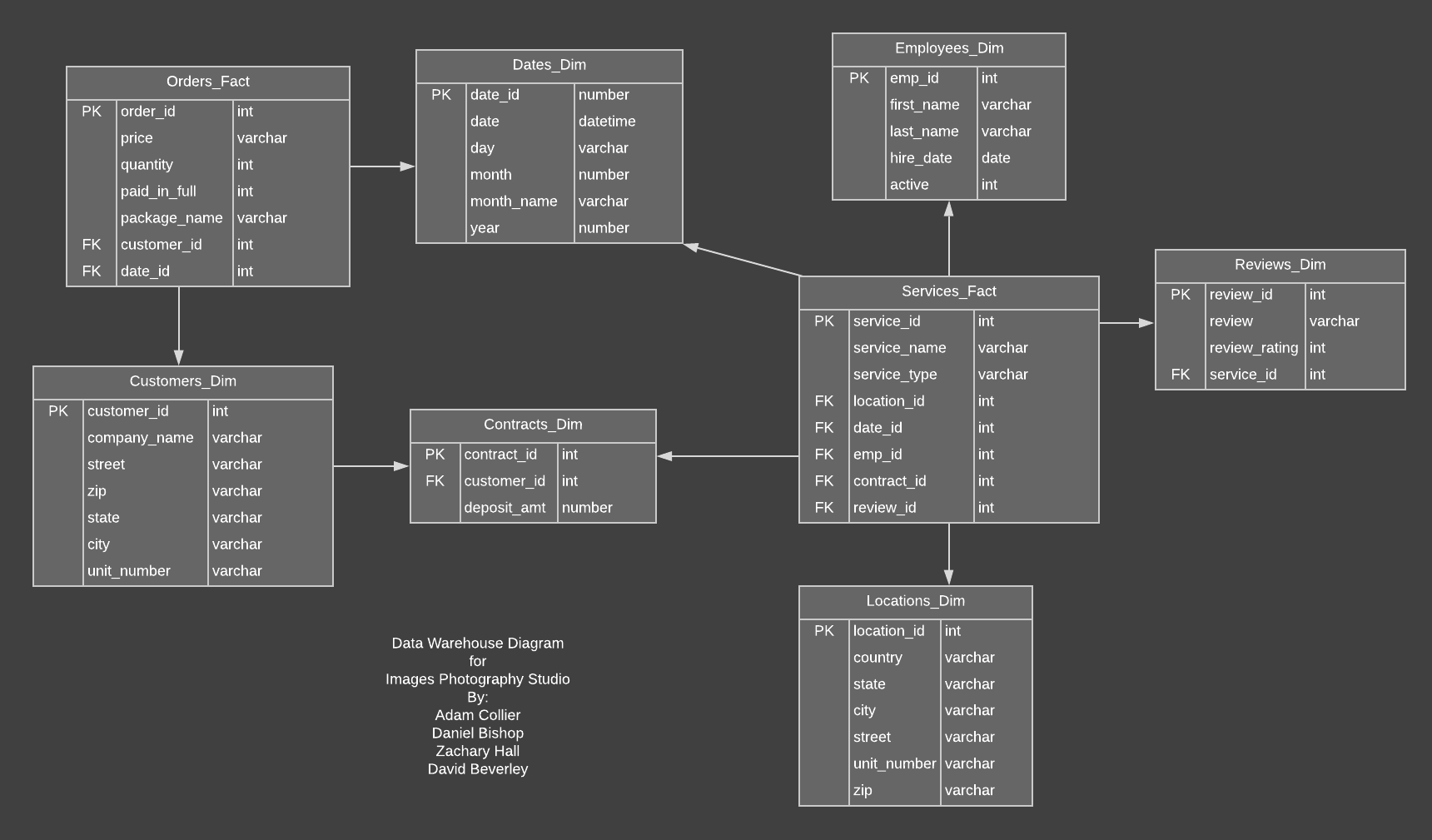
Data Warehouse design  
and ETL PROCESS

for Images Photography Studio

By: David Beverley, Adam Collier, Zach Hall, and Daniel Bishop

**Dimensional Data Warehouse Design:**



**Design explanation and reasoning:**

The design above is the planned structure for the data warehouse for Images Photography Studio. In this design there are two fact tables that will be used for running reports of the company’s data.

The Orders Fact table is for reporting on statistics relating to the packages that a customer order. With this table reports like the popularity of packages based on the customers city or state, analyses on average package cost, and average deposit amount per city can be ran for statistical analysis.

The Services Fact table is for reporting statistics on the services that are held by the company. This table will enable reports on employee statistics like their popularity per location, the average rating (1 to 5) for the type of service, what services are popular in each location, service popularity depending on the time of the year.

**Extract-Transform-Load Plan:**

Extraction:

The initial extraction for the production database will be a full extraction, for later extractions the database will notify when there are updates to the database and the updates to the data warehouse will be extracted based on company needs. When extracting the data, tables will need to be revised to better support the needs of the data warehouse.

The orders table will drop the description, special instructions fields, balance, and balance\_paid attributes, they are not useful in the needs for the data warehouse. The packages and order\_packages attributes for the package name, price, and quantity will be consolidated into the orders table.

The customers table will drop the first and last name attributes, this is because there is not useful reporting that can be performed based on individuals’ names.

The contracts table will drop the signed attribute.

The services table will drop the service\_time and end\_time attributes. The service\_types tables type will be consolidated into the services table. The assist\_emp\_id will be stored in the services table to further simplify the data.

The roles, emp\_roles, emp\_availability, pictures, and proofs tables will not be included for any part of the extraction to for the data warehouse. No notable or useful data for statistical reports can be extracted from these tables.

Transformation:

When transferring a productional database to a data warehouse, some of the data may need to be cleansed, validated, and derived from other values.

No values in the data warehouse will need to be derived from multiple entries in the transactional database.

Some of the values will need to be cleansed due to the allowance of null values in the transactional database. The customers table will have any null values in the company\_name attribute converted to “Individual”.

Loading:

The first-time data is loaded into the data warehouse, it will be a full load of the current transactional database. From then on, the data will be loaded incrementally base on the company’s reporting needs. Once the data is loaded into the data warehouse, test modeling views will be created on the fact tables and reports will be ran on the system to check the consistency of the data and to ensure that the data transition was successful.

**Is Mongo DB the right choice for the Data Warehouse?**

MongoDB and NoSQL in general does not make much sense as a solution to the data warehousing for this photography company. One of the main draws of MongoDB is that it stores high-availability, non-normalized data for exceptionally large databases.

By our calculations there will not be enough data to justify the need for a non-relational model. The data set will measure in the thousands of rows over tens of columns, not nearly approaching the millions of data pieces in most examples of MongoDB databases even over the course of months of data collection.

With as little data as there will be created a relational database would still be adequately efficient in terms of query speeds. A well-designed relational database would have minimized joins to pull necessary data for long-term planning reports. The data was originally designed to be stored in a highly normalized form as photography packages and data about events and employees will always be regular outside of rare custom orders.

The last major advantage is the high availability of MongoDB databases, but with the frequency of the reports being generated this is not as essential as other elements of the CAP theorem. While these reports may be run weekly or even monthly, the database will not need to be touched daily, or at the very least if a day is missed it would not cause severe problems with the data being stored and calculated on.

In either case the process for moving data from the production database into a Mongo DB style data warehouse would be similar to a standard ETL process. First, we would create a data warehouse in the manner shown on the previous page in SQL. After which we then export the SQL data warehouse into a Mongo DB collection using tools provided by both parties. In the end of the process, the data would be stored on a Mongo Db cluster in a warehouse like manner to be kept and used for analytical purposes. However, as stated above, we will not be going this route due to the lack of substantial benefits for our use case.