Full Database Copy

Goal: We have a huge shared database structure with hundreds of tables. We need to surgically cut it into smaller mini pieces per corporation. The end goal is a structure that we could move pieces around and manipulate as needed (manual load balancing, add, edit, delete, move to cold storage, duplicate, copy, etc.)

Step 1: We need a file that has all of the create table pieces. This will be called “create\_new\_solar\_system.cfm” We need to combine all of the mini sql files into a single page that is wrapped in ColdFusion code.

1. Open the mini .sql files.
2. Copy and paste into the .cfm file
3. Go though and format each SQL entry.
4. Add <cfquery> tags around SQL create table code.
5. Fix spacing for readability.
6. Remove all ` (little slanted single quote thingys)
7. Remove the last ; (semicolon) for the end of each SQL statement.
8. Remove the AUTO\_INCREMENT=x value from each SQL statement.
9. Repeat the process for the next table.

Step 2: We need to add indexes to all of the date features in the new database. This is a feature that didn’t exist in the older, shared database structure. We only want to index dates. We want to leave the date/time stamps alone for now. Just the plain dates. Here are the steps…

1. Look at each table and see if there are any dates.
2. If yes, go to the section where the “keys” are set. These are the indexes.
3. Let’s say we found a date that looks like this as part of the create table SQL:
	1. balance\_action\_date date DEFAULT '0000-00-00',
4. Add this to the keys section.
	1. KEY balance\_action\_date (balance\_action\_date)
	2. You may need to add commas and spaces as needed.
5. Check for multiple dates per table. Each date needs its own key.
6. Repeat the process for the next table.

Step 3: We need to copy certain look-up tables and populate dummy records. This is step 1 of populating the new database. This is not corporation specific data yet, this only deals with basic pieces that are needed by all systems.

1. additional\_customer\_types
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-7)
	4. This table is a sub look-up table for the additional\_customers table. Basically it deals with how or what kind of an additional customer is this? Ship to, bill to, cosigner, end user, etc. Additional customers may be assigned to invoices, quotes, and stock/units (serialized inventory).
2. additional\_customers
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a sub of the main customers table. It is connected to the additional\_customer\_types table. These additional customers are used for invoices, quotes, and stock/units as additional customer tie-ins. Optional one-to-many relationship. One special note, this table will tie to a customer id but will physically stamp the data as things happen. So in a way, it is a sub of the main customers table but it is also a standalone piece to help record the story as things happen. For example, say the ship to address was set to something at a certain point in time, then later on set to something different. This table will record that info as it happened.
3. adjustment\_pos
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is connected to the main purchase\_orders table. This is virtually a special list of what update PO’s have been put in place. This helps with the update inventory counts section.
4. afb\_corp\_settings
	1. solar system level
	2. normal insert
	3. only insert dummy record (id=1)
	4. This is the AFB Snow Owl theme settings. This includes all kinds of colors, special links, special verbage, options, theme settings (including stuff for footers, headers, and special tables).
5. afb\_payee\_settings
	1. solar system level
	2. normal insert
	3. only insert dummy record (id=1)
	4. This is how you control the dynamic colors. The reason it is called payee settings is because they will allow each person to setup their own look and feel. It technically still ties to the Snow Owl theme but it allows them to customize look, feel, flow, etc to make them more comfortable. This has a lot of hex colors and css styles. This also holds special custom links per person, for instance you can setup your own favorites (8-10). You have quick links to pages that you can change.
6. allinv
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is the meat of the stock/unit section. Stock/Units are one of the main 12 application player groups. Basically, they are big ticket items and/or serialized inventory. The allinv table has 147 different fields that hold data. This table is linked to the following other tables: inventory\_types, makes, models, sub\_inventory\_types, title\_accounts, allinv\_history, payee (multiple different ways), invoices, payroll\_status, allinv\_subs, store\_location, allinv\_final\_numbers, allinv\_asset\_types, allinv\_floorplan, customers, customer\_types, purchase\_by\_types, stock\_photos, tie\_in\_flex\_grid, cms\_media\_main, custom\_documents, web\_price\_settings. This one is one of the original tables. Huge piece of the original puzzle. This is where it all started.
7. allinv\_ages
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table holds dates, ages, and stock/unit info. The purpose is to track stock/units from location to location. Basically, where does it go as it gets moved around. This table is a sub of the main allinv table. It is also connected to the store\_location table.
8. allinv\_asset\_types
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a sub of the main allinv table. It deals with whether or nor you, as a company, own the inventory or if it is on consignment. This is related to floorplan stuff as well as a feature called the manager’s checkbook (slush fund stuff).
9. allinv\_final\_numbers
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a sub of the main allinv table. It deals with calculated final sales numbers once a stock/unit is sold. This helps keep the math totals in one place for quick reference. Once again, this only deals with sold stock/units. Units still in inventory don’t play in this table. All units in inventory get their math calculated on the fly until they are marked sold.
10. allinv\_floorplan
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a sub of the main allinv table. It deals with floorplan activity. Stock/Units may be floored or held by outside banking entities. This is basically a section that allows stock/unit titles (ownership of the unit) to be traded, bought, and sold at one or more banks for monies and/or funding. This table is also connected to the title\_accounts table.
11. allinv\_history
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a sub of the main allinv table. This is where system-maintained history records are stored per stock/unit. This includes adds, edits, deletes, location moves, sales, and other history related records. This table is also connected with the payee table and the history\_flags table.
12. allinv\_internet\_status
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-4)
	4. This table is a sub of the main allinv table. This is a small look-up list that deals with showing the stock/units over the web to customers. This has values like Inv, Sold, and Take Off. There is automatic code that flips this value based on the main stock/unit status.
13. allinv\_payments
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a sub of the main allinv table. This table helps record stock/unit payments made toward a base price and/or floorplan activity. This provides support for stock/unit payables. This table is also connected with receipts, receipt\_line\_items, payee, and allinv\_floorplan.
14. allinv\_status
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-4)
	4. This table is a sub of the main allinv table. This is a small look-up table that helps the stock/unit have a main status. These are values like Inv, Sold, and Deleted. This determines the inventory status of a single stock/unit.
15. allinv\_sub\_types
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-5)
	4. This table is a sub of the allinv\_subs table. It includes basic categories for classifying sub dollar amounts to stock/units (main allinv table). This table includes categories like: Outside Repairs, Parts & Internal Invoices, Shipping, Miscellaneous, and Inventory Adjustments (slush fund stuff).
16. allinv\_subs
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a sub of the main allinv table. It is used to hold one-to-many relationships between a single stock/unit and possible other details. Some of the details include: outside repairs (expense/receipts), part & internal invoices (internal invoices or parts tickets per stock/unit), other shipping & miscellaneous fees, and inventory adjustments (slush fund or manager’s checkbook) options. The table holds the category of the sub (tied to the allinv\_sub\_types table), the dollar amount, the id reference (expense/receipt or invoice number), the date, and small notes about the sub item.
17. app\_status\_history
	1. Cluster level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table monitors the number of server resets. For the record, a server or dedicated box is a cluster in our world building analogy. This table catches a date/time stamp, a note, and an IP address of when the system got reset and/or restarted. This could be a manual reset or an automated action. Basically, a system monitor table.
18. app\_types
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-14)
	4. This table contains a numeric reference id number for each of the main 12 application player groups. This includes things like deposits, invoices, PO’s, expense/receipts, user-maintained balance sheet items, stock/units (serialized inventory), customers, vendor/payees, employee/users, parts and general inventory items, elements of time (calendar and scheduling), and quotes. This table is related to the following tables: tie\_in\_flex\_grid, tie\_in\_flex\_grid\_titles, tie\_in\_flex\_grid\_history, cms\_meida\_main, custom\_documents, payee, and time\_sub\_assignments. As a note, most of these tables may be connected to one or more main application player groups hence the need for knowing which one to connect to. An application type id and an individual player id make a unique combo that identify a single individual or data object. For example: deposit # 111, invoice # 444, PO # 555, etc.
19. balance\_sheet\_archives
	1. solar system level
	2. normal insert
	3. only insert dummy record (id=1)
	4. This one is not being used right now. It’s original purpose was to store a historical record pdf snapshot for financials. There isn’t really a plan to use it currently, but it is still a good idea for the future.
20. balance\_sheet\_history
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a sub of the blanace\_sheet\_items table. Its job is to store and record the system-maintained history records for individual user-maintained balance sheet items. This includes adds, edits, and other history related records. This table is also connected with the payee table and the history\_flags table.
21. balance\_sheet\_items
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is the main user-maintained balance sheet item table. User-maintained balance sheet items are one of the 12 main application player groups. This where special financial pieces, not held elsewhere in the system, are stored. It could be an asset (we own or have), liability (we owe), or equity piece (worth). This could be things like investments, loans, equipment, buildings, vehicles, paid in capital, distributions, retained ernings, depreciation, etc. Basically, anything that needs to play into the picture and is not handled elsewhere in the application. Hence the name user-maintained. This table is also connected to: financial\_categories, financial\_groups, financial\_group\_subs, store\_location, payee, balance\_sheet\_subs, balance\_sheet\_photos, payee, balance\_sheet\_history, tie\_in\_flex\_grid, cms\_media\_main, and custom\_documents.
22. balance\_sheet\_photos
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a sub of the main balance\_sheet\_items table. It holds individual photos, images, and scans for user-maintained balance sheet items. Up to 100 images per individual user-maintained balance sheet item. This table is used for documentation or visual reference of some kind. This table is also connected to the payee and main balance\_sheet\_items table.
23. balance\_sheet\_subs
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is a sub table that supports the main balance\_sheet\_items table. A user-maintained balance sheet may contain sub line items and/or activity entries. This table holds the line items and/or activity details. In a financial realm, this allows the different user-maintained balance sheet items to flex and/or change values over time. Think of the line items like a virtual entries or actions that get applied to the bigger user-maintained balance sheet items. This is how you bump dollar values up or down depending on what is needed. This table is also connected to: balance\_sheet\_items, payee, deposits, invoices, receipts, and purchase\_orders.
24. bank\_check\_specs
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is a sub table for banks. It holds check printing and alignment information. The table stores \_x (horizontal) and \_y (vertical) measurements (coordinates) for printing pre-printed check stock. The coordinates are then passed to an Adobe Flash widget at runtime for actual web based check printing. The flash widget then puts the correct fields at the correct places and then prints the check using a normal inkjet or laser printer. This table is tied to the banks and check\_types tables.
25. banks
	1. Solar system level
	2. Special insert - we need three different records to be copied in to the database. They are id’s: 1, 7, and 50. 1=Dummy, 7=No Bank Assigned, and 50=Void
	3. This table creates a virtual bridge between deposits (monies coming in) and expense/receipts (monies going out). At least one bank is required to run the system. Unlimited number of banks per corporation. Numerous check writing options available. This table is connected to check\_types, bank\_check\_specs, deposits, and receipt\_payments tables.
26. biotrack\_api
	1. solar system level
	2. normal insert
	3. only insert dummy record (id = 1) – currently doesn’t have dummy record
	4. This is specific to some state compliance API’s. This records which corp, invoice and response to make sure things are synced up correctly between our system and the state compliance system.
27. biotrack\_api\_methods
	1. solar system level
	2. normal insert
	3. only insert dummy record (id=1) – currently doesn’t have dummy record
	4. This is specific to some state compliance API’s. Not really sure what this does in addition to the biotrack\_api table, but it is meant to help make sure things get synced up correctly between our system and the state compliance system.
28. cart\_favorite\_categories
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1) - make sure that id 1 is called “General Category”.
	4. This table is sub section of the my cart favorite buttons. The special my cart favorite buttons are precoded buttons to help with sales and getting items into the shopping cart. Tons of options including basic searches, quick add to cart, custom settings, group options, tiered pricing, and even possible tie-ins with recipe/builds (kits, groups, and preset list of items). This category table acts as the category or grouping for the buttons. By default, all buttons are setup under a general category. However, if you want to change the display and sort order, you could create as many top level categories as you wish. Say you were a restaurant or deli. You may want top level categories like: Starters, Entrées, Combo Meals, Desserts, and Drinks. You could then have a number of sub my cart favorite buttons under the main categories. This table is a sub of the my\_cart\_favorites table. It is also connected to the main payee table.
29. chart\_of\_account\_type
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-18)
	4. This table is a sub of the main chart\_of\_accounts table. As a bit of history, we originally headed towards traditional accounting terms and standard chart of accounts (late 2006). Then because we developed the system in stages, we only ended up using special containers (virtual accounts) in three main sections. Because we only use special user defined containers in the three areas (deposits, expenses, and balance sheet items) we changed the name form standard chart of accounts to expense types, deposits types, and balance sheet types. We found that we could cover some of the other areas in the system by using math, other in-line categories, and time-based concepts. Currently, only three of the 18 original account types are in use.
30. chart\_of\_accounts
	1. Solar system level
	2. Full id insert
	3. Special id list. We need three different records to be copied in to the database. They are id’s: 1, 649, and 834. 1=Dummy, 649=Void, and 834=Not Assigned
	4. This is the main table that holds special account types. In traditional accounting, this is called the chart of accounts. In adilas, we strayed from the standard formula due to our development cycle. Instead of allowing users to create whatever chart of accounts they wanted or needed, we limited access to application pieces as they got finished and released. In late 2006 we were headed toward traditional chart of accounts. Then we launched code that dealt with banks, deposits, and expense/receipts. Because we only had those pieces done, we didn’t want to call them chart of accounts, for fear of not actually having either a P&L (profit and loss or income statement) and/or a balance sheet at that time. So, internally, we changed the names to expense types and deposit types. The names stuck and they have been in play since 2006. Eventually we added the income statement in late 2007 and then worked on the first rounds of the balance sheet in late 2008, early 2009. At that point, early 2009, we added a third type called balance sheet types. Currently these three account types are the only ones in play inside the adilas system. This table is also connected to deposit\_line\_items, receipt\_line\_items, chart\_of\_account\_type, financial\_categories, financial\_groups, and financial\_group\_subs tables.
31. check\_request\_types
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-4)
	4. This table is a sub of the normal expense/receipts section. This table deals with a thing called a check request. Basically, it is a temporary expense/receipt that needs approval. Once it gets approved, it will become a normal expense/receipt. Kinda like a quote that may become an invoice, but this is for an expense. If yes, it will become a real expense. If no, it can be voided out without any problem. It can even go through multiple approval and denial processes if needed. This table is tied to the main receipts table, receipt\_types, and multiple tie-ins to the payee table (aka vendors, payees, users, employees, etc.)
32. check\_types
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-7)
	4. This table is a sub of the banks table. This table allows a single bank to have different check printing options. This could be handwritten (non computerized), pre-printed check on the top, pre-printed check in the middle, pre-printed check on the bottom, or other triplicate (old school) pre-printed options. As a note, these settings are sent to the adilas check write app (small Adobe Flash widget) for web-based check printing. There is also a table called bank\_check\_specs that will hold the alignment and field positioning for the check write settings. Checks are part of the payment portion for expense/receipts and check requests.
33. cluster\_corporations
	1. cluster level
	2. normal insert
	3. insert dummy record (id=1), we also need access to corp 22 (Adilas, main admin control) and corp 53 (test site).
	4. This is the cluster controller table. This is the table that keeps track which corporations are actively on it’s box. This is also how we would lookup to see which data source would be used for the different corporations. For example, say you have corporation 500, you could look up it’s name, alias, corp key, and which database it connected to from this table. Say you are at a hotel and you are the person behind the counter and you want to look up “Jimmy” and find that he is in room 113. If a corp moves to a new box we can just flip that corp to inactive.
34. cluster\_payee
	1. cluster level
	2. normal insert
	3. insert dummy record (id=1)
	4. This is a master list of which world each user belongs to (home planet). This cluster level actually will help us bridge payees across solar system across worlds. This is a common id for the box, verses an individual id per database (solar system). This is primarily used for login and switching corps where the normal payee table will be used for the smaller transactions.
35. cluster\_payee\_to\_corp
	1. cluster level
	2. normal insert
	3. insert dummy record (id=1)
	4. This is how we can tell which person has access to which corp. Originally this was going to be very similar to the payee\_to\_corp\_to\_permission table but we ended up taking out the permissions out of it and making it a general switch. This will hold the cluster payee id and the corp ids with a general status of active or not. It is a high level of which worlds you can touch.
36. cms\_media\_file\_types
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-15)
	4. This table is a sub of the cms\_media\_main table. The acronym CMS stands for Content Management System. A CMS deals with files, documents, and digital archives and digital catalogs. This table helps to categorize what kind of file and/or document is being archived. These are things like: text, PDF, spreadsheet, image, video, audio, web, other file type.
37. cms\_media\_history
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is a sub of the cms\_media\_main table. This table is the system-maintained history record for all media/content (digital files) actions. This includes notes about adding, editing, uploading, and other history related records for the file and/or actual documents that get archived. This table is tied to the cms\_media\_main, payee, and the history\_flags tables.
38. cms\_media\_main
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is the main media/content table. This table holds the reference and catalog information for all digital files, archives, and physical uploads. The media/content section is quite extensive and allows all kinds of interactions with documents and digital files. Think of this section as a giant digital filing system or document library. You may catalog, reference, upload, link to, or cross-tie to any existing piece of data in the system. Files may be held on a general level, as part of the 12 main application player groups, or clear down at the individual level within a group. Tons of options. This table is also connected with app\_types (all 12 system player groups), cms\_media\_file\_types, cms\_media\_history, cms\_media\_references, and the payee table. As a note, most corporations (worlds – level 5 stucture) are configured to use a media/content server that is outside the normal data path. This gets into a level 2 structure called a galaxy (aka links servers).
39. cms\_media\_references
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-5)
	4. This table is a sub of the cms\_media\_main table and media/content section. The media references (this table) deal with where the files are stores. We allow local references (files stay on your computer or a network drive), remote references (files are stored on any other cloud or web based drive such as DropBox, SkyDrive, Google Drive, etc.), and physical uploads (these files are physically stored on adilas media/content servers). Lots of options.
40. colors
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-34)
	4. This table is a sub of the main allinv (stock/units) table. It contains a list of basic colors for big ticket items or serialized units. As a note, this table is used in searches and to get the item stocked into the system. Once in the system, the user may alter the color to fit their needs. For example, say you have a stock/unit that is black, red, and white. They, the user, would select one of the colors from the list and finish adding the unit to the system. Once the item is in the system, they could edit the stock/unit details and change the color to black/red/white or whatever. Basically, this list of colors helps get the ball started but is not actually tied to the stock/unit once it is entered into the system. Mostly just a look-up table to start the process.
41. common\_features
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (numerous records) – currently all tie-in on makes, models, and stock/units. Go ahead and grab everything with a corp id of 1. We then may have to grab corp specific stuff as well.
	4. This table is a sub feeder of the main allinv table (stock/untis). It deals directly with the makes and models tables. A common feature is created, tied to an inventory type, and then tied together with a certain make and model to create a pre-formatted list of common features for specific models. Say you were selling a vehicle. Some common features might be: power windows, sunroof, cruise, lift kit, tow package, etc. If you were selling a trailer, you might use: v-nose, ramp – fold up, ramp – slide in, side door, spare mount, extra D-rings, etc. On a technical level, here are the tables that play together: makes, models, model\_to\_common\_features, inventory\_types, and corporations.
42. condition\_types
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-5)
	4. This table is a sub of the allinv (stock/units) table. A condition type is something like new, near new, used, or rental. Small little feeder table. Used mostly on reports as a filter and while stocking in a unit into inventory.
43. contact\_types
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-22)
	4. This table is a sub of the customer\_log table. A customer log is a user-maintained note and/or follow-up event for a single customer. The word customer inside of adilas is tied to a setting. This makes it dynamic and allows you to change it to meet your needs. This could a customer, client, associate, student, patient, member, etc. Anyways, the contact types help to set the flavor for each customer log note. These are things like: phone call, left message, in person, text message, email, fax, appointment, follow-up, meeting, delivery, etc. Lots of options to choose from. These values will show up as part of the customer log entries. Make as many as you wish to keep tabs on your customers. These values are also searchable when searching the log entries. Tied to the customers and customer\_log tables.
44. corp\_to\_inventory\_types
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a linker or joiner table between corporations and inventory types. The inventory types are main high level categories that a corporation and/or business may interact with. The corporation administrator may turn inventory types on/off at will. Once turned on, certain parts of the application began to open up to allow you to play in those areas. If certain inventory types are turned off, no actions are allowed in those areas. Inventory types are things like: parts and general inventory, labor/services, trailers, vehicles, toppers, specific units, rentals, and balance sheet items. Technical table connections are: corporations and inventory\_types.
45. corporations
	1. Solar system level – these are the worlds within the solar systems.
	2. Full id insert
	3. Special id insert - we need three special id numbers copied into the database. They are: 1, 22, 53. 1=Dummy, 22=Adilas, and 53=playground testing
	4. The corporations table is the table that stores word level settings and details. The world level is level 5, right after solar systems (level 4 – databases), clusters (domains and servers), galaxies (linked servers such as data servers and media/content servers), and universe (internet and all of adilas). This table has close to 200 (and growing) different fields that hold settings, options, and instructions for how and what your world will look like and be. This table contains the id numbers that are cascaded out to all of the smaller tables to create unique identifiers for your world. This is the table that stores your world settings. Technically, this table is connected with well over 150+ of the other tables in the database. Any table that contains the column corp\_id is a sub part of this table. Once again, this is the world level (level 5). We allow you to configure your system as you see fit. Just imagine, there can be many worlds inside of a solar system (database). There can be many solar systems inside of a cluster (server). And the list goes on… This is adilas! Your data, your world, your way! Dream it up, we’ll help you wire it up!
46. country
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-195)
	4. The country table has a listing of different countries. By default, the country code is set to United States. This table is tied to banks, store\_locations, and corporations. This table may become more important as we go forward. Currently it doesn’t do a whole lot and is defaulted to the US.
47. custom\_dates
	1. solar system level
	2. normal insert
	3. no dummy record
	4. This is a special table to help us hold data specific values that were not fields in normal tables. For example for sub inventory if we need specific sub attributes that are dates, we will store the dates here. These tables are dynamic and can be tied to any other tables. This is kind of like our generic date storage spot. Currently this is only used for sub inventory but was built like this to expand to other areas as well, kind of like a big brother to flex grid. As a note this table goes along with custom\_numerics and custom\_text.
48. custom\_docs\_to\_corps
	1. Cluster level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. Custom documents may be assigned to corporations at the cluster or domain level (level 3 – aka individual server level). This table creates a link between the custom\_documents (Excel, PDF, web forms, paperwork, etc.) and which corporations or worlds have access to the custom documents. This table is basically the assignments or settings per world. By default, no custom documents are assigned. However, this table allows for assignments to be made per corporation (per world) and per main 12 application player groups (deposits, invoices, PO’s, expense/receipts, customers, stock/units, vendors, items/parts, elements of time, employee/users, quotes, and balance sheet items). As a unique feature, these assignments may store up to 10 hardcoded values as needed. This is used for items or data not stored inside of the normal adilas application. The special settings may also hold special mapping values to help with dynamic data feeds. Very flexible. Technically, this table is connected to custom\_documents, app\_types, and corporations.
49. custom\_documents
	1. Cluster level
	2. Full id insert
	3. Need all data and id’s (record 1-38)
	4. This table hold the actual custom document names, references, code pages, and instructions. Once a custom document is added to the system, it then needs to be assigned to different corporations or worlds. By default, just because a custom document exists, doesn’t mean that any world or corporation may use it. The document still needs to be assigned to the world or corporation using the custom\_docs\_to\_corp table (world level assignments). Once that connection is made, the system will automatically show links and options to use the custom documents from inside of adilas. Often times, custom documents are Excel files, Adobe PDF, or other special web forms or reports. These custom documents usually pull data from the system and show the custom options in special pre-programed or special formatted layouts. The custom documents pull the data based on dynamic coded values. For example: Image a state required form for a vehicle registration. The PDF will be supplied by the state and has a specific format and layout. We over the PDF form with data fields and then populate all known fields on the special form. We then allow for open entry on other parts of the form. This is just one small example of what a custom document is. Once the document is populated, it may be printed, saved, and even pushed back up to the adilas media/content server for later reference. This section is how we accomplish custom invoices, state and federal forms, special reports, custom mapping options for labels, and much more. Contact an adilas rep and/or consultant for more information. Custom documents start as low as $100 per document. What do you need? Let’s help you get it wire up! Lots of untapped potential here…
50. custom\_numerics
	1. solar system level
	2. normal insert
	3. no dummy record
	4. This is a special table to help us hold data specific values that were not fields in normal tables. For example for sub inventory if we need specific sub attributes that are numerics, we will store the numbers and decimals here. These tables are dynamic and can be tied to any other tables. This is kind of like our generic numeric storage spot. Currently this is only used for sub inventory but was built like this to expand to other areas as well, kind of like a big brother to flex grid. As a note this table goes along with custom\_dates and custom\_text.
51. custom\_text
	1. solar system level
	2. normal insert
	3. no dummy record
	4. This is a special table to help us hold data specific values that were not fields in normal tables. For example for sub inventory if we need specific sub attributes that is text, we will store the text here. These tables are dynamic and can be tied to any other tables. This is kind of like our generic text storage spot. Currently this is only used for sub inventory but was built like this to expand to other areas as well, kind of like a big brother to flex grid. As a note this table goes along with custom\_numerics and custom\_dates.
52. customer\_contacts
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a sub of the main customers table. A customer/contact is a one-to-many relationship between a single customer record and multiple different points of contact. For example: Say you dealing with a business called XYZ Construction. XYZ would be the main customer record. Then, say you wanted to record a shipping address, a billing address, one or more owners, a manager, and your favorite sales rep. All of those extra pieces would go under different customer contacts (manually create one-to-many relationships). You can record job titles, businesses, first and last names, addresses, city, state, zip, phone numbers, emails, web addresses, notes, etc. For a different example: Say you were dealing with an individual, a customer contact might be: a wife, a husband, one or more parents, children, care giver, doctor, baby sitter, accountant, etc. Basically, any other piece of the puzzle that needs to be tied to a single customer record. These are called additional contacts.
53. customer\_credit\_types
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-5)
	4. This table is a sub of the main customers table. It holds small credit options such as cash only, credit limit (fill in the blank), credit hold, and unlimited credit. These values help when building new invoices and shopping carts for customers. If a limit is set, the application will prompt and show existing outstanding amounts due (accounts receivable). Mostly this table helps when setting up a new customer or changing their settings later on.
54. customer\_history
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a sub of the main customers table. This is where we record system-maintained history records on what happens to different custom records. This includes adds, edits, new sales, new quotes, new log notes, new additional contacts, new elements of time (scheduling and calendar events), etc. All of these actions are recorded behind the scenes as you interact with the customer record. It helps tell the story from the systems point of view. This table is tied to the main customers table, payee, and the history flags table.
55. customer\_log
	1. Solar System level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is tied to the main customers table. A customer log is a user-maintained note or log entry. You can add as mnay of these as you want to per customer. An example might be: you had a phone call with them, they came in and visited with you, you may want to copy & paste an email they sent to you, etc. The nice thing is once you put it on their record it becomes available for everyone else to see. This table is tied to the main customers table, the contact\_types table, and the payee table.
56. customer\_photos
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is tied to the main customers table. This is where you get to document photos, scans and images related to your customer. Up to 100 photos/scans per customer. This helps you document any piece that need to be part of the story. For example: contracts, photos, license numbers, business cards, projects, etc. This table is connected to the main customer table and the payee table.
57. customer\_queue
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a sub of the main customers table. A customer queue is virtually a waiting room or who’s next to be serviced. The queue can actually be dynamically named to meet your needs. It also has a number of other settings primarily dealing with time, checking in, and waiting to be serviced. From the queue there are direct link right to calendar entries and shopping cart pieces. This table is tied to the main customers table, invoices, quotes, elements\_of\_time, and payee tables.
58. customer\_queue\_types
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-10)
	4. This table is a sub of the customer queue table. It contains actions and statuses of the queue. This would be things like started, waiting, invoiced, quoted, in progress, closed, etc.
59. customer\_types
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is a sub of the main customers table. This is how you classify your different customers. You can create as many categories as you’d like. This really helps with reports, filtering customers records and possible variable pricing structure. For example some possible customer types might be: member, non-member, wholesale, retail, VIP, contractors, dealers, etc. As a note, these customer types are pulled into the shopping cart once a customer is selected. They are also passed on to the my cart favorite buttons section in case you have special pricing tiers.
60. customers
	1. Solar system level
	2. Normal insers
	3. Only insert dummy record (id=1)
	4. This is the main customers table. This is one of the 12 main application player groups. The word customers is a dynamic setting inside of the system which allows you to change it to whatever you may need. For example: customers, contacts, clients, members, associates, students, patients, etc. This section can be standalone all by itself or potentially connected with tons of other pieces inside of the system. For example this section may connect to invoices, quotes, eCommerce, elements of time (calendaring & scheduling), accounts receivable, deposits, and standalone CRM functionality. CRM stands for Customer Relationship Management. Other possible connections might be photos, media/content, additional contacts, user-maintained logs, statements, histories, outstanding invoices, sales histories, custom documents, flex grid tie-ins, and anything you need to track information related to your customers. This table is tied directly to customer\_types, customer\_credit\_types, payee (salesperson assignments), customer\_log, customer\_history, customer\_photos, customer\_queue, invoices, quotes, elements\_of\_time, tie\_in\_flex\_grid, and many others. This can get pretty deep.
61. daily\_visits
	1. solar system level
	2. normal insert
	3. only insert dummy record (id=1)
	4. This is special for people who check-in and how many times during the day. This is a custom table that isn’t used very much but the idea is that a check-in method is stored (mag stripe, DL number, etc.) and not necessarily tied to a customer.
62. db\_field\_settings
	1. solar system level
	2. Normal insert
	3. Only insert dummy record (id=1) currently no dummy record.
	4. The deeper we get the more we see that people want custom labels and what not. This table helps give us the ability to alias the entire database. This is kind of like the model view controller for the entire system. Currently this has only been lightly built out in customers but has huge potential for the whole system. This allows you to name it, alias it, show/hide it, sort order, rules (required/num chars, special instructions, etc). The concept is cool but the implementation needs some major loving (only cascades to 2 pages).
63. departments
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a sub of the payee’s table. Basically, departments is an internal reflection of your company’s organization or structure. A department is directly tied to a location and then tied to people who work within that department. The word location is very generic. An example of a department may be: sales, admin, techs, accounting, etc. This table is directly tied to store\_location, payee, and payee\_to\_departments tables (actual assignments & tax settings per person). A department is required for any digital time clocks and payroll options inside the system.
64. deposit\_history
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record
	4. This table is sub of the main deposits table. This is where we record system-maintained history records that deal with each deposit. This could be adds, edits, updates, removals, verifications, etc. All of these records are recorded behind the scenes to help with the audit trail. This table is connected to the main deposits table, payee, and history\_flags tables.
65. deposit\_line\_items
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a sub of the main deposits table. This is where all of the deposit information is held and stored. This deals with money types, amounts, tie in to customers, invoices. Also on an accounting level, how they are categorized for financials. This is the meat of the deposit. This is a one-to-many from deposits to line items meaning the number of line items per deposit is unlimited. Inside of adilas, the default deposit line items comes from invoice payments that need to be deposited. This creates your connection between customers and invoices and monies coming in. This table is connected to the main deposits table, store\_location, chart\_of\_accounts, money\_types, allinv (stock/units), and invoice\_payments tables.
66. deposit\_photos
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is a sub of the main deposits table. This table allows for photos, scans and images for deposit related transactions. This is very helpful if there are weird or special circumstances happening or you need to document something, or any transaction that needs some extra babysitting. Up to 100 photos per deposit. This table is connected to the main deposits and payee tables.
67. deposits
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is the main deposits table. This is one of the 12 main application system players. Deposits are how you record monies coming into the bank. By default, most times deposit line items are from invoice payments. But they could be from other sources such as loans, investments, refunds, profit sharing, commissions, etc. The goal of the main deposits table is to create the container to put all of the other pieces into. This table is very simple and basically connects itself to the banks, the date, the amount and small notes. The bulk of the work is done through the deposit line items table. This table is directly connected to banks, deposit\_line\_items, and payee tables.
68. direct\_deposit
	1. solar system level
	2. normal insert
	3. no dummy record
	4. This is new from Will Hudson (Adilas developer). Basically it is storage information for banks and what is needed to make bank calls. Banks need certain information in certain formats and this is where we store that. This is currently an extra paid function that has to be turned on to be able to use.
69. elements\_of\_time
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is the main elements of time table. This is one of the 12 main application player groups. Elements of time deal with anything that is time related such as: scheduling, calendaring, rentals, to-do lists, special notes, events, billing for time, tracking time, etc. Elements of time is very versatile and may be customized to meet your needs. Without getting into great detail, we let you name it, set what functionality you want, and then use it. Unlimited possibilities. By default this section has 4 different views of the data: calendar view, details view, grouped view, and time slot view. The goal with elements of time is to eventually connect any person, place, or thing together with time as the wrapper. This table connects to store\_location, customers, payee, invoices, quotes, time\_templates, time\_action\_logs, time\_photos, time\_sub\_assignments, time\_sub\_comments, time\_sub\_dates, time\_sub\_gps, time\_sub\_payroll, time\_sub\_pool\_assignments, time\_sub\_pools, time\_sub\_reminders, and time\_sub\_sign\_offs tables.
70. elements\_of\_time\_history
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a sub of the main elements of time table. This is where we record the system maintained history notes relative to time pieces. This includes adding, editing, updating, changing, invoicing, quoting, assigning, etc. All these records are done behind the scenes but help to provide a story of who touched what and when things happened. This table is tied to elements\_of\_time, payee, and history\_flags tables.
71. error\_reporting
	1. cluster level
	2. normal insert
	3. no dummy record.
	4. This is what errors happen per box. These are exceptions, undeclared variables, timeouts, logic errors, etc. This table gets hit if an exception is thrown. This basically physically tracks cold fusion errors that are happening.
72. external\_transactions
	1. solar system level
	2. normal insert
	3. no dummy record
	4. This is tied to the direct\_deposit table. This is the transactional data based on the direct deposit information.
73. fed\_state\_allowances
	1. Cluster level
	2. Full id insert
	3. Need all data and id’s (records 1-numerous)
	4. This table is on the cluster level and is a sub look-up related to payroll, different federal and state withholdings. Each year new records are added to this table to be in compliance with current tax withholding laws and regulations. This table is connected with the payroll\_frequency table.
74. federal\_tax\_tables
	1. Cluster level
	2. Full id insert
	3. Need all data and id’s (records 1-numerous)
	4. This table contains just federal tax withholding look up values. These records are also modified every year to remain in compliance with current federal tax rates, laws and regulations.
75. financial\_categories
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-11)
	4. This table is level 2 of 5 levels for how we create our mappings for income statement and balance sheet types. The level above this is called money\_destinations which basically maps to either the income statement or balance sheet. Below this level we also have financial\_groups, financial\_group\_subs, and chart\_of\_accounts tables. Inside of adilas we don’t use the standard terminology called chart of accounts. Please see the chart of accounts table description for why we altered this.
76. financial\_group\_subs
	1. Solar system level
	2. Full id insert
	3. Special query – we need all records that match a corp id of 1.
	4. This table is level 4 of 5 that create our financial category groupings. As a reference it goes money\_destinations, financial\_categories, financial\_groups, financial\_group\_subs (this table), and chart\_of\_accounts.
77. financial\_groups
	1. Solar system
	2. Full id insert
	3. Special query – we need all records that match a corp id of 1.
	4. This table is level 3 of 5 for creating financial category groupings. As a reference it goes money\_destinations, financial\_categories, financial\_groups (this table), financial\_group\_subs, and chart\_of\_accounts.
78. general\_table
	1. Cluster level
	2. Full id insert
	3. Need all data and id’s (record 1-9)
	4. This table helps switch between test and live modes and also contains yearly social security withholding limits. This one just holds general information. It’s main purpose is switching between test and live and holding miscellaneous numbers for payroll.
79. hardcoded\_parts
	1. Solar system level
	2. Full id insert
	3. Special query – wee need all records that match a corp id of 1.
	4. This table holds a master list of internal system items that you get just for signing up with adilas. We call these hardcoded parts. The word parts inside the system is a dynamic name. For example it could be called: parts, SKU’s, services, products, items, etc. By default each company gets between 15-20 hardcoded values they can turn on and off as needed. Some of these values have special code behind them to help them with special functionality in the system. Some of the parts you get by default are: labor, shop supplies, freight, shipping, discount, fee, verbage, other, etc. Technically this table does eventually connect out the parts table but it is mostly used for pre-loading the system.
80. help\_files
	1. Cluster level
	2. Full id insert
	3. Need all data and id’s (records 1-400+ & growing)
	4. This table helps us to be able to put help files on individual pages within the system. This table is a sub of a table called web\_pages. Each help file has an id and a password and can contain whatever information we determine is helpful including graphics, links, texts, etc.
81. history\_flags
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-13)
	4. This table is a sub of all of the system history tables inside the system. This is basically a look-up for what kind of system history is being recorded. This deals with new/add, edit, update, void, assign, lock, unlock, delete, etc. This table is a sub of many of the history tables. Search for the word history\_flag\_id to see which one it connects to.
82. ifta\_vehicle\_mileage
	1. Solar system
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a special table for the trucking industry. IFTA stands for International Fuel Tax Association. Basically this is GPS coordinates for which trucks drove in which states for what dates, times, and mileage. This is used for reporting. This is a sub of the allinv table.
83. ifta\_vehicle\_mileage\_interface
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a special table we use as a prep step for the IFTA vehicle mileage table. Basically we get GPS coordinates uploaded into our system in a CSV file. CSV file is a comma separated values file or a poor man’s database file. We then take the CSV file, push it into a table, run validation on it and then start sending it where it needs to go.
84. inventory\_types
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-10)
	4. This table controls what options are available at a corporation or world level. If the items are not assigned to your corporation or world you cannot even play in those sections. Some of the main inventory types are: parts and general inventory, service/labor, balance sheet items, specific units, trailers, toppers, vehicles, etc. Lots of specific code is cascaded throughout the system based on these inventory types and how they can play (financial for P&L and balance sheet). This table is related to corp\_to\_inventory\_types, allinv, parts, and corporations tables.
85. invoice\_api
	1. solar system level
	2. normal insert
	3. only insert dummy record (id=1)
	4. The idea behind this table is to help track API calls and responses to help with state compliance and data synchronization. Currently this is being used to deal with Leaf Data and Metrc.
86. invoice\_history
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is sub of the main invoices table. This is where we record the system-maintained history for all invoice actions. This includes adds, edits, updates, removals, voids, change in customer assignment, assignments to time, etc. All of these actions are recorded behind the scenes and available within the site if needed. This table is tied to the main invoices table, payee, and history\_flags tables.
87. invoice\_payments
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a sub of the main invoice table. This is where you record how things were paid for, change due, what kind of payments they were, etc. As a huge note, this ends up creating the basis for your deposits. This is also how we figure out accounts receivable based for a customer by doing simple math between invoice amounts and sum of invoice payments. As an additional note, these are the things that get connected to merchant processing and credit card processing pieces of the puzzle. If merchant processing is used additional reference numbers are included. No actual credit card information is stored, it is all passed to secured gateways, but this is the piece that holds the reference. This table connect to the main invoices table, payee, money\_types, deposit\_line\_items, and merchant\_settings tables.
88. invoice\_photos
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a sub of the main invoice table. This is where you record any photos, scans and images related to invoices. This could be payment information, signed documents, contracts, refunds, exchanges, etc. Up to 100 photos/scans per invoice. This table is connected to the invoices and payee tables.
89. invoice\_status
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-7)
	4. This table is actually a sub of elements of time. It sounds like it is related to invoices but it helps prepare things for billing for time. As an example some of the categories might be: no charge, invoiced, needs to be invoices, quoted, etc. Once an element of time has been set to this status, it can then be pulled up and grouped so you can see what has been invoiced and what has not been invoiced. There is then a section that allows you to add multiple items to the cart at a single time. The invoice status will then be changed behind the scenes based on actions done in the cart.
90. invoice\_types
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-9)
	4. This table is a sub of invoice, quotes, and the shopping cart. The invoice types help the system know how to treat this type of transaction. Each invoice type has different rules and requirements it has to follow. For example a transfer invoice requires two locations (a to and a from), a customer invoice requires a customer to be assigned, and an internal ticket requires a stock/unit number. Other possible invoice types are counter sales, quotes, transitional invoices, wholesale carts, etc.
91. invoices\_[corp\_id]
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is the main invoice table. This table is how you sell things out of the system. What you sell depends on your company. It could be products, services, labor, stock/units, time, etc. We even allow mixed tickets, which means you can mix and blend any of the pieces on a single invoice. Invoices are one of the main 12 application system player groups. All invoices get created through the shopping cart or through online eCommerce. Based on the invoice types, invoices can do different things. Don’t get stuck on thinking invoices are only used for customer sales, invoices can also be used internally as a tool in the system. For example, transfer invoices (moving items from place to place), internal repair invoices (taking products of the shelf and applying it to a bigger stock/unit), updating inventory counts and shrinkages, and even transitional pieces such as layaway or work orders. This table is connected to store\_location, customers, allinv, payee, invoice\_types, po\_invoice\_line\_items, invoice\_payments, invoice\_photos, invoice\_history, elements\_of\_time, and tie\_in\_flex\_grid tables.
92. makes
	1. Solar system level
	2. Full id insert
	3. Special query. We need everything that has a corp id of 1
	4. This table contains a list of makes for stock/units or big ticket items. For example some makes for a vehicle might be: Ford, Chevy, Dodge, etc. For trailers it might be: Big Tex, Hallmark, Pace, Parker, etc. You get the idea. Basically, if we don’t have the make already in there you get to add it so that it ends up becoming a make and model combo for the stock/units. This table is connected to inventory\_types, models, and allinv tables. If you are using specific units, the name make may be altered as a setting.
93. managers\_time\_clock
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is part of the manager’s time clock permission. Digital time clocks are used by employees for payroll. Employees cannot fix their own timecards, this requires an admin or manager to make a change. The manager’s time clock permission requires two pieces. One, that the permission itself gets turned on and then two, which departments the person is assigned to be a manager over. This is the assignment table for the permission. For example: say we had 4 departments for our business. Let’s also say that we wanted Sally to be over sales and technicians. This table would hold Sally’s assignments to be over those 2 departments. This table gets connected to corporations, departments, and payee tables.
94. master\_corporation\_list
	1. Cluster level
	2. Full id insert
	3. Special query. We need two records. We need id numbers 1 and 22. 1=Dummy and 22=adilas
	4. This table is at one of the highest levels. This is how we control which corporations are assigned, this is the master list of corporation id numbers. A corporation is basically a world. Another part of this table lets us know which cluster (aka box, domain, or server) the corporation is located on. This table is connected to individual corporation tables throughout the whole universe.
95. master\_stock\_list
	1. Cluster level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table holds a master list of stock/units and which worlds and clusters they belong to. This is actually used for GPS tracking. Each item gets its own id number, then when we get bulk GPS data back, this table helps us know where the stock units are located at within their worlds. We then disseminate GPS data to the worlds and clusters based on this master list. This table is connected to allinv and corporations tables.
96. merchant\_settings
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is where we hold the settings and assignments to the different merchant processing gateways. This table combines location info, bank info, corporation info, and then gateway settings. In English that means, how can I connect to credit card processing. Any sensitive information is encrypted for storage. Merchant settings may be applied by location or globally per company or corporation. This table is connected to invoice\_payments, merchant\_types, store\_location, banks, and corporations tables.
97. merchant\_types
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-10)
	4. This is a sub of the merchant settings table. This table contains a list of the different credit card merchant processing gateways that we can connect to. For example: USAePay, Authorize.net, MAXX Merchants, PayPal, FirstData, etc.
98. model\_to\_common\_features
	1. Solar system level
	2. Full id insert
	3. Special query. We need all records
	4. This table is a sub piece of stock/unit makes and models. It actually links up a model to things that are called common features. The real usage of this comes when you are adding in a new stock/unit. Once you select a model, this automatically knows which common features have been assigned to that model. This becomes the basis of your unit description. For example: say you had a model set up with 10 different common features, when you add that model, those 10 features will automatically be populated into the description. This table is tied to models and common\_features tables.
99. model\_to\_usage\_classes
	1. Solar system level
	2. Full id insert
	3. Special query. We need all records
	4. This table is also a sub of the stock/unit makes and models. This one deals more with the utility or the why and what particular models are used for. This part of the puzzle has been only partially developed. You can record functionality on the model level, but it doesn’t actually populate into a description, it is more for your information. Originally the goal was if you wanted to search by functionality we could return models back to you that matched that functionality. This may be a future project that could be further developed. This table is tied to models and usage\_classes tables.
100. models
	1. Solar system level
	2. Full id insert
	3. Special query. We need all records
	4. This table is a sub of the stock/unit section. All stock/units must have a make and model. The models table connects a make, a sub inventory type, and model descriptions all in one place. If you are using specific units, the name model may be altered as a setting. This table is connected to makes, inventory\_types, sub\_inventory\_types, allinv, models\_to\_common\_features, and models\_to\_usage\_classes tables.
101. money\_destinations
	1. Solar system levels
	2. Full id insert
	3. Need all data and id’s (records 1-3)
	4. This table is the top level for a 5 level structure for financial categories. Basically this table says, do I show up on the income statement or the balance sheet. Then the other 4 levels figure out the rest of the sorting or groupings. For your reference here are the 5 different levels: money destinations, financial categories, financial groups, financial group subs, and chart of accounts. Please see the chart of accounts section for some of our naming convention.
102. money\_types
	1. Solar system level
	2. Full id insert
	3. Special query. We need all records
	4. This table classifies what kind of transaction is being dealt with. This table ends up connecting out to invoice payments, deposit line items, expense/receipt payments, checks, etc. Some examples of money types are: cash, check, AMEX, Discover, MasterCard, VISA, ACH, internal, Debit Card, wires, EFT, etc.
103. my\_cart\_favorites
	1. Solar System
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table holds all of the settings for pre-programmable buttons called my cart favorites. A my cart favorite button is actually a small adobe flash widget that can hold custom settings behind the scenes. Buttons may be simple, photo, grouped, they can contain colors, search parameters, custom settings, tiered pricing, they can be nested, tied to recipe/builds, they can be tied to smart groups (which can be tiered pricing), basically, you get to create your own interface buttons. You get to name them, categorize them, organize them, nest them and so on. These features can work as standalone buttons for the shopping cart or they can be used in a split cart interface with the buttons on one side and all interactions taking place on the other side. Classic point of sale type scenario. This table is tied to cart\_favorite\_categories, payee, parts, recipes, and customer\_types tables.
104. part\_categories
	1. Solar system level
	2. Full id insert
	3. Special query – we need two records. They are id’s: 1 and 2. 1=Dummy and 2=Internal Use
	4. This table is tied to the main parts table. This is how you categorize your general inventory items. You may have as many categories as needed. These categories are used for sales and they also translate into financial reports. These same categories end up being used on your P&L for revenue, cost of goods sold, etc.
105. Parts
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is the general parts and inventory table. This is one of the main 12 application system player groups. The word parts is a corp-wide setting which means it can be changed to whatever values you need. For example: parts, items, SKUs, products, services, widgets, etc. This category can hold both quantitative items and items that we consider to be unlimited, such as labor and services. These items are brought in through the PO section, they are then sold on invoices and quotes through the shopping cart. Special web and eCommerce settings can be applied so that they can be viewed by people outside of the secured environment. Parts and general inventory are also the basis for my cart favorite buttons, recipe/builds. This is also the primary item that holds the barcodes, RFID tag tracking values, and other references. This table connects to payees, part\_categories, units\_of\_measure, po\_invoice\_line\_items, recipe\_line\_items, my\_cart\_favorites, parts\_history, parts\_photos, tie\_in\_flex\_grids, quote\_line\_items, tax\_categories, web\_price\_settings, and inventory\_types tables.
106. parts\_history
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a sub of the main parts table. This is where we record the system maintained history notes and records. This deals with adds, edits, cost changes, part number changes, status changes, web setting changes, etc. This table is connected to payees, history\_flags, and the main parts table.
107. parts\_photos
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a sub of the main parts table. This is where are the photos, images, and scans for your parts and general inventory items may be held. Some bonuses of this particular table are that the same photos may end up being some of your photo buttons under my cart favorites, these photos are also used directly under your eCommerce and web presence sites. These photos may also be used on your invoices and quotes where you show pictures per line item. Up to 100 photos or scans per item. Also by way of a note, if you add image captions, they may end up being part of your search engine optimization (or SEO). This deals with how text values are found after a search engine crawls your site. This table is connected to the main parts and payees tables.
108. pay\_type
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-7)
	4. This table is a sub of the payroll section or payee\_to\_department table. This table contains a list of different pay types such as: hourly, salary, commission, hourly + commission, salary + commission, etc. Once the setting is applied to a payee through a department assignment, it is used every time that payroll is calculated. The system will try to look up digital time clocks, invoices for commissions, etc. to match up with your pay type.
109. paycheck\_history
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is tied to the main paycheck\_stubs table. The paycheck stubs actually hold the payroll withholdings information, gross net, social security, commissions, pay rates, departments, etc. This table is the history portion of that. It keeps track of the system maintained history for when it is created, edited, tied to an expense, voided, etc. This table connects to paycheck\_stubs, payee, and history\_flags tables.
110. paycheck\_stubs
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is where the main payroll or paycheck stub information is recorded. This table will end up creating your year to date sums and values if you use the adilas payroll system. Adilas payroll automatically deals with digital timecards for hourly employees, commissions on multiple levels if assigned, salaries, state and federal withholdings, pay frequencies, and other employee payroll settings. This table also includes options for sick pay, vacation pay, bonuses, games, draws, non-tax withholdings, child support, 401K’s, etc. Each time payroll is run a brand new check stub is created. Once it is created it can then be tied to an expense where you can physically print out the check or setup an EFT directly to your employee’s bank account. This table is connected to payee, departments, receipts, payee\_to\_department, and paycheck\_history tables.
111. payee
	1. Solar system level
	2. Full id insert
	3. Special query – we need 6 records. We need id numbers: 1, 2, 3, 6, 1821, 2036. 1=Dummy, 2=Brandon, 3=Adilas Setup, 6=Steve, 1821=N/A System, and 2036=Stock/Unit. Some of these ids have special code attached to them.
	4. This table has a dual purpose. This same table holds two different groups of the 12 main player groups. One, it holds all vendor information per corporation or world. Two, it also holds all employee/user information. As a note, employee/users are technically bigger than worlds because one employee/user may be able to span, visit, or work in multiple worlds. Vendors are corporation or world specific. The difference between the two deals with permissions inside of the system. Only employee/user’s get permissions, vendors do not. The reason that both groups, vendors and users, are on the same table, deals with how PO’s and expense/receipts are recorded. Both of these main player groups require a payee, which technically could be either a vendor or an employee/user. The payee table connects to almost every table inside of the system. This deals with who is allowed to play, who has played, and who will play inside the system. If you want to find all of the tables that deal with the payee tables, look for one of these two values: payee\_id and vendor\_payee\_id. As a further technical note many of the tables even double and triple join to this table multiple times. For example: who created it, who approved it, who authorized it, who edited it, etc. Here are some of the 50 million tables that this one might be connected to: all history tables, all photo gallery tables, allinv, cms\_media\_main, customer\_log, customer\_queue, customers, deposits, elements\_of\_time, invoices, managers\_time\_clock, my\_cart\_favorites, parts, paycheck\_stubs, payee\_login\_history, payee\_personal\_settings, payee\_to\_corp\_to\_permission, payee\_to\_department, payee\_type, po\_invoice\_line\_items, po\_payments, purchase\_orders, quotes, receipts, recipe\_line\_items, reoccurring\_invoice\_actions, store\_line\_items, tie\_in\_flex\_grid, etc.
112. payee\_history
	1. Solar system
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is tied to the main payee table and this is where we record all of the system maintained history for who adds, edits, or modifies a single payee record. As a note, payees come in two types, vendor payees and employee/user payees. Also as a reminder an employee/user is bigger than any corporation or world. If needed we can also pull multi-corporation histories on employee/users.
113. payee\_login\_history
	1. Cluster level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table deals with who accesses the box, server, or domain level. This is called the cluster level. Basically it time stamps who comes in, what IP address they came in on, and if they log out it also captures the date/time and the outgoing IP address. This table is at the higher cluster or box level.
114. payee\_permission\_categories
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-9)
	4. This table has the main categories for how we assign and categorize our permissions. This table has a history back to 2006. The classic homepage is built off of this structure. Some of the values are things like system search, system basics, system time and requests, system assets, system liabilities, system reports, system maintenance, and system management. These were our main categories when we started this project. This table is connected to the payee\_permission\_list table.
115. payee\_permission\_list
	1. Cluster level
	2. Full id insert
	3. Need all data and id’s (all records & growing)
	4. This table is the main permission id list. Permissions are categorized, given an id, given a human readable name, and a description of what the permission entails. The backend developers cascade the required permission checks to the appropriate pages. Each page within the system then checks to see if the corporation, the user, and the permission assignment match before dealing with or using any of the pages. This is a triple check that is done on each page. Permissions have color codes, availability to be either a main permission that will show up on the classic homepage, or a sub permission that deals with something deeper in that section. There are even some permissions that would be considered a negative, or limiting permission. Permissions have been categorized into sales, manager, accounting, corp admin, specialty, system admin, etc. There are close to 140+ permissions and counting. This table gets connected to the payee\_permission\_categories, payee, and payee\_to\_corp\_to\_permission tables.
116. payee\_personal\_settings
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table deals with the way an individual employee/user (payee) interacts with the system on a daily basis. This includes things like: your default location, default interface or homepage, default expense payment method, default PO type, default invoice type, etc. Future growth is anticipated in this table. This table is tied to payee, money\_types, po\_types, and invoice\_types tables.
117. payee\_photos
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table handles all payee photos and is a sub of the main payee table. This includes vendor payees and employee/user payee photos. These are images, scans, photos, documentation, etc. for these individual players. Up to 100 photos/scans per payee. There is some additional security added to this information as it may include employee and HR records and documentation.
118. payee\_to\_corp\_to\_permission
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is a joiner table between the permission list and the payee/user assignments. This is who has access to what. This table will link a corporation to a user and to a permission. As a note, users are bigger than corporations or worlds, because they may play in multiple areas, thus the 3-way connections. This allows for a person in one corporation to have specific permissions and to have more or less in a different corporation. Making each corporation’s user permission assignments completely independent and specific to that corporation. This is the table that gets checked on every single page to make sure that the user has access to what they are requesting. This table is connected to corporations, payee, and the payee\_permission\_list table.
119. payee\_to\_department
	1. Solar system
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table will create the connection between a payee/user and a department. This is where all of the payroll information is held for a specific user based on the department the user is assigned to. A user may even be assigned to multiple departments as needed. This table holds pay rates, pay frequencies, commission rates, withholdings, tax settings, etc. This table goes fairly deep and is where payroll gets all of its information. Lots of settings and switches. This table is attached to the payee, departments, work\_shift, and payroll\_frequency tables.
120. payee\_to\_stores
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table connects a vendor/payee to a store or location. This table is somewhat of a holdover table from prior versions. As a fun history note, this table dates back to a time before the payees were corporation specific. For example: say the vendor Wal-Mart just existed as a single vendor and anyone was able to use it. However as we started branching into other corporations using the adilas system, we had to make vendors become world specific to maintain security and integrity. This also dealt with account numbers per vendors and per locations. This table only has minimal legacy pieces that still exist. All new transactions are done completely corporation specific.
121. payee\_type
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-5)
	4. This table holds the classifications that help us determine the different payee types. Some of the different payee types are vendors, employee/users, and customers. The two main payee types on this table are used to differentiate between vendors and employees. This determines who can log in, who gets permissions, and who can be assigned to a PO or expense/receipt. As a special note on customers, this usually deals with a customer that bought something (aka – invoice or quote) and also needed to be paid something. The system is able to convert a customer record into a payee record so that they can exist in both places. For example this could be used on a refund or a return. This table is a sub of the main payee table. Once again, the only payee type that can login and use the system are the employee/users.
122. payroll\_frequency
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-9)
	4. This table deals with the payroll frequency or the rate at which payroll is calculated. This includes values such as daily, weekly, bi-weekly, semi-monthly, monthly, quarterly, semi-annually, and annually. This is a sub of all payroll functionality. This setting is held in all look up tax tables and also in the user to department settings. Technically this table deals with these tables: payee\_to\_department, paycheck\_stubs, fed\_state\_allowances, federal\_tax\_tables, and state\_tax\_tables.
123. payroll\_status
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-7)
	4. This table is tied directly to invoices that are tied directly to payroll. By default any new invoice gets a payroll status of “new sale”. The person in charge of payroll is then able to approve it, okay it, mark it as pending, and control when it advances into the payroll section. Only invoices that are okayed for payroll are pulled into the current payroll section.
124. photo\_upload\_history
	1. Cluster level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a master list of all photos and scans that get uploaded into the system. Its main job is to record sizes and compression rates on a global or box level. Individual photos are held within smaller player group specific tables. This table mostly helps system administrators monitor who is attempting what uploads, dates, types, and sizes.
125. po\_history
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is a sub of the main purchase orders table. This is where we record all of the system maintained history notes dealing with adding, editing, line item changes, removals, voids, verifiying, updates, payments, etc. This table is connected to the purchase\_orders, payees, and history\_flags tables.
126. po\_invoice\_lines\_[corp\_id]
	1. Solar system level
	2. Normal insert
	3. Special query. This will be handled in the add new corp function.
	4. This table is how all of your inventory is tracked. In prior releases there was a giant shared table called po\_invoice\_line\_items. This was the first table that we had to split out and make corporation and world specific, thus the X in the table name. What we found was that this was the table where a single corporation could have hundreds and thousands of records just for their world. To handle that, we made this table corporation specific. This table holds hands with the main purchase orders table and the main invoices table. This is the meat of all of your PO and invoice line items. By having this table combined it can deal with positives (PO’s) and negatives (invoices). This also allows for a great deal of versatility in inventory tracking allowing you to go backwards, forwards, multi-location, multi-category, etc. Traditionally many systems will record quantities at the item level. That only works if you are trying to pull current inventory counts that are typically for a single location, or a predetermined number of locations. The way adilas works is that we allow it to be a one-to-many dynamic table where we can handle any number of locations, any date specific requests backward or forward, and we keep all of the usage on one table for quick reporting. This table has many cause and effect relationships that extend all the way out to revenue, cost of goods sold, receivables, payables, inventory levels, usage, etc. This table finds its way to your main financial documents in differing ways. This includes querying this table to get details and sums for your income statement (P&L) and your balance sheet. This table connects to the corporations, payee, purchase\_orders, po\_types, invoices, invoice\_types, customers, part\_categories, parts, tax\_categories, units\_of\_measure, store\_location, inventory\_types, and allinv tables.
127. po\_payments
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is tied to the main purchase orders table and holds payments made towards the balance due on a PO. This table, even though directly related to PO, is technically accessed through expense/receipt line items. The application is setup to watch for a valid PO number to be entered into the PO field on the expense/receipt line items. If yes, then the application will apply the payment in the background and create the connection between the two. This creates the full circle connection between where the money came from and what it goes to. This is a part of the payables section that is automated. Without getting super detailed there are 3 ways to pay for a PO. They are, 1: manually enter the PO number into the PO field on the expense/receipt line item, 2: using the automated section of the payables homepage for paying PO’s (able to pay PO’s in bulk), and 3: there is a PO builder that allows you to add multiple PO’s to a single expense from inside of the expense/receipt line item page. This 3rd options is sometimes used if you are paying for both PO’s and other items to the same vendor and want to make a single check/payment. This table is connected to the main purchase orders table, receipts, and payee tables.
128. po\_photos
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is tied to the main purchase orders table and holds the photos, scans and images for PO’s. Up to 100 photos per PO. This is used for documentation, invoice scans, statements, or any other related documentation for a PO. This table is tied to the payee and purchase\_orders tables.
129. po\_types
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-7)
	4. This table is a sub of the main purchase orders table. This is how we classify a PO so that it gets special functions and handling. Inside of adilas we use PO’s for all kinds of inventory based functions, not just for acquiring products. Many of the PO types are tied directly to independent permissions within the PO section. Also many of the standard naming conventions for these PO types are dynamic corp-wide settings. This may be somewhat random but if you end up working in PO’s you may need to reference the help file under the main add/edit PO section. There is over a paragraph per PO type that also explains some of the ins, outs, and special functionalities for each type. Some of the types are things like: update PO’s (update inventory quantities), internal build PO’s (internal mini manufacturing), basic live PO’s (vendor specific), basic request PO’s (orders), special live PO’s (mixed vendors), and special requests PO’s (mixed vendor orders). This table is a sub of the main purchase orders table. It is also used as a special look up on the PO/invoice line items table.
130. purchase\_by\_types
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-4)
	4. This table is a sub of the allinv table for stock/units. This small lookup table helps determine whether an item was purchased, traded in, or consigned and by whom. As a small side note, this also deals with a thing called the manger’s checkbook (slush fund).
131. purchase\_orders
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is the main purchase orders or PO table. A PO is one of the 12 main system player groups. Inside of adilas we use PO’s in many different ways to help with inventory functionality. This includes bringing items into the system, setting up labor and services, updating inventory counts, internal builds and mini manufacturing, etc. By default PO’s also create an automatic payable that shows up on your financials. PO’s may be paid off as a single PO or in bulk using expense/receipts (another application player group). PO line items end up creating the incoming side of inventory and all usage reports. This is where your costs are set. It is also possible to use the PO section to do ordering for products that have not yet been delivered. This table is tied to the store\_location, payee, po\_invoice\_line\_items, po\_payments, po\_types, po\_history, po\_photos, adjustment\_pos, receipt\_line\_items, and tie\_in\_flex\_grid tables.
132. quote\_history
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a sub of the main quotes table. This is where we record system maintained history notes such as: creating new quotes, updating items, and converting to invoices. This table is tied to the main quotes table, history\_flags, and the payee table.
133. quote\_line\_items
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a sub of the main quotes table. This table holds all of the line items that are assigned to a shopping cart and then saved into a quote. The line items may contain parts, general inventory items, labor, services, stock/units, elements of time, etc. Line items may also be shown and hidden as needed. One of the cool things about quotes is they may be restored to the cart and then modified from inside the shopping cart as needed. You can save quotes as often as they need to. This table is tied to the main quotes table, parts, store\_location, payee, tax\_categories, and allinv (stock/units) tables.
134. quote\_photos
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a sub of the main quotes table. This is where any sort of documentation, potential contracts, or requirements may be stored as images, scans or photos. Up to 100 photos per quote. This table is connected to the main quotes table and the payee table.
135. quotes
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is the main quotes table. Quotes are one of the 12 main system player groups. Quotes serve a few different purposes. They may be a quote as you imagine it, such as something quoted to a customer but not purchased yet. They may be saved as adobe pdf’s and emailed, printed, etc. Quotes may also be just a saved version of the shopping cart. This is great if you need to do something else in the middle of a transaction or the customer is coming in later and you want to prep the deal. Quotes are also highly used through the eCommerce section either as normal quotes or as web orders (no payments). Then these quotes are restored back to the cart and finalized when the person shows up to pick up the product or make payments. Quotes may also be saved as many times as needed. If you want a new version of the quote, you can use the save as new version feature and create as many versions as needed. This table is tied to the quote\_line\_items, quote\_history, quote\_photos, invoices, payee, store\_location, invoice\_types, customers, and allinv (stock/units) tables.
136. reasons
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-11)
	4. This table is a sub of the digital timecards section. The reasons are basically a preset list of why people would be clocking in and out. This includes things such as: start work, going home, work, health, lunch, family, school, sick pay, vacation pay, holiday pay, etc. This table is directly tied to the time\_cards table.
137. receipt\_history
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is sub of the main receipts table. The word receipts inside of adilas is sometimes expressed as expense/receipts or E/R’s. This table contains all of the system maintained history notes such as: adding, editing, printing checks, reimbursements, changing vendors, paying for PO’s, paying for stock/units, floorplan options, etc. This table is tied to the main receipts table, history\_flags, and the payee table.
138. receipt\_line\_items
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a sub of the main receipts table. This is how you categorize what is being paid for. At least one item is required per expense/receipt but it may contain as many as needed. The different line items have an amount, are each assigned to a location, and are assigned to a thing called an expense type (which is a virtual chart of accounts). Expense/receipt line items also have options to be tied to invoices, PO’s, stock/units, floorplan payments, or attributed to a stock/unit as an outside repair. This section is the work horse behind expense/receipts and what they can do. If you have multiple locations expense/receipt line items may also be split between different locations based on preset percentages. The expense/receipt line items section plays a deeper financial piece of the puzzle as well and is tied to cost of goods sold, expenses and other balance sheet items. This table is connected to the main receipts table, store\_location, chart\_of\_accounts, allinv, allinv\_subs, purchase\_orders, and invoices tables.
139. receipt\_payments
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a sub of the main expense/receipts table. This table deals directly with transactions to banking entities and monies going out. We also use this table to record payments that are not yet paid, called payments on account. This section has a lot of check printing options to allow you to print directly from the web, based on permissions. If a check is printed and has a problem, you are able to reprint the check. A special history note will be recorded and a flag will appear if reprinted. This section is also directly related to how you reconcile bank accounts. These payments also allow us to track monies still owed (payments on account or splits), reimbursements, and internal transactions. All kinds of different payment options are available from here such as: cash, check, EFT’s, ACH’s, Visa, Mastercard, debits, wires, transfers, etc. This table is connected to the main receipts table, money\_types, banks, and payee tables.
140. receipt\_photos
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a sub of the main expense/receipts section. This is where you get to document any sort of invoice statement, physical receipt, payments made, or documentation for the expense/payment. Up to 100 photos, images, and scans per expense/receipt. This table is tied to the receipts and payee tables.
141. receipt\_splits
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a sub of the main receipts table. The whole goal of this table is to allow a system so that one payment can satisfy multiple smaller subs. For example: say you’ve bought 12 things from a vendor at different times, all on account, and under their own expense/receipts. Finally you want to pay the vendor off but you only want to write one check. What this table or section allows for is virtually being able to piggy-back all of these smaller expense/receipts onto one master expense/receipt you can then pay. Technically, this is called a one-many-many relationship. Another good analogy of this is like a truck that carries other cars to a dealership. Each little car is almost like its own little miniature expense, they could each stand independently. But the truck that carries all of them, holds these independent pieces and can wrap them together in one main package. This table connects receipts to receipts and it is also connected to the payee table.
142. receipt\_types
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-7)
	4. This table is a sub of the main receipts table. This deals with how we categorize or classify the different receipt types. For instance: normal expenses, check requests, blank check requests, approved check requests, declined check requests, and voids. All expense/receipts are normal unless you are using the check request system. The check request process preps the expense but does not allow for any payment information until the request gets approved (requiring an additional step or approval). By the time everything finishes, it ends up being a normal expense receipt, having had varying statuses throughout the process. Check requests are great for corporations that have multiple locations where decision makers need to approve payments before checks are written and payments are processed. As a side note we generally discourage check requests unless they are helpful for your corporation.
143. receipts
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is the main expense/receipts table. An expense/receipt is one of the 12 main system player groups and possibly one of the deepest. Of all the player groups, expenses are the only ones that can stack, carry, or be combined with its own player group. In the simplest form an expense/receipt is monies going out of the bank and how they are categorized. All expense/receipts are tied to a payee vendor or user. As it gets more advanced, the options open up tremendously. Expense/receipts are used to pay off PO’s, floorplan payments, stock/unit payments, outside repairs (sub expenses applied directly to a stock/unit), payments on account (splits), reimbursements, balance sheet item transactions, check requests, payroll, etc. An expense/receipt may have one or more expense types (aka chart of accounts) applied to the different line items. They are also capable of having an unlimited number of line items, payments, payment methods, and may be split between locations. This table is connected to the payee table (multiple times), receipt\_types, receipt\_line\_items, receipt\_payments, receipt\_history, receipt\_photos, allinv, allinv\_payments, tie\_in\_flex\_grid, banks, chart\_of\_accounts, store\_location, etc.
144. recipe\_history
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a sub of the main recipes table. Often we say recipe/build to talk about all recipes. Some other common phrases for this type of feature are: groups, kits, packages, deals, al-a-cart menus, and pre-built lists of options. There are multiple different kinds that run either on the invoice side or the PO side for miniature manufacturing or internal building up of products. This table helps us keep track of all of the system-maintained history items, such as adds, edits, and other modifications made to the recipe. This table is connected to the recipes table, payee, and history\_flags tables.
145. recipe\_line\_items
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is a sub of the main recipes table. Recipes allow for multiple items to be grouped or combined together to create an outcome. Whether it is trying to sell or make something in bulk7. Inside the system, we call these ingredient or output items. This table helps hold settings for price structures, cost structures, custom descriptions, show/hide options, tax categories, quantities, and other things that might be part of the recipe/build, kit, package, etc.
146. recipe\_types
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-4)
	4. This table contains the different types of recipes we can do. By default there is a thing called build and sell and that is kind of like a quick invoice group similar to a package or kit. The next kind is a build and hold which is tied to a more of a PO type of build. This is used for a miniature manufacturing (transferring one type of product into another, i.e have nuts and bolts and paint them and then sell the painted bolts.). The last kind is called build and build. This has not been fully developed yet but this is when we wanted to build one on top of the next on top of the next and wanted to tie it to stock units. Instead of miniature manufacturing it would be full on manufacturing. This would be like building for more of a stock unit type level (i.e getting tires and rims and seats and fames and putting them all together to build the car).
147. recipes
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. The main table for the recipes. This is where we get to name things and setup what it yields, what kind it is and any sort of notes. This also has options for bar codes in case you want to get to something quickly. This is the main container that ties to the recipe\_line\_items as ingredients for the recipe. (i.e. if I want to make big mac combo, on recipe\_line\_items I would have the hamburger, fries, drink but I would get to it by coming to the main recipe called “big mac combo”)
148. relationship\_types
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-7)
	4. These actually tie into the flex grid. Every flex-grid tie in allow tie ins such as parent, child, brother, sister, N/A, detail, and other. As a note if you choose other you are required to type in what the relationship is. Once again these are used on the flex grid records.
149. reoccurring\_invoice\_actions
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. What this does is this is the transaction on how the reoccurring invoices happen. This table will record which one was copied, which one was created and what are the dates. So behind the scenes there is a master invoice number that will be duplicated over and over again. This shows what the transaction that happened against that record. For example if I build a master invoice number 500 I can then duplicate that invoice over and over again in a reoccurring fashion.
150. reoccurring\_invoices
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is where we setup that master invoice number. This is also where we get to setup the reoccurring time frame or period (if it is yearly, bi-yearly, monthly, etc).
151. smart\_group\_assignments
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is part of the my cart favorite buttons. One of the different types of buttons is called a smart group. The smart group is basically tiered pricing. Within the setup there are two things that a smart cart button can handle. 1. Rules – how to change price based on qty. 2. Assignments – which tables are assigned to this set of rules and tiered pricing matrix. This is the table to assign items to buttons. This is an unlimited number of assignments you can make per matrix.
152. smart\_group\_rules
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is also tied to my cart favorite buttons. This is the rules or where you actually get to create the rules and how they get used. This includes starting quantity ending quantity, description changes and the pricing options. Some of the pricing options are per or grand total (reverse math). You can setup an unlimited number of rules per matrix.
153. solar\_system\_names
	1. Cluster level
	2. Normal insert
	3. Insert dummy – may also want to insert if we have a generic cluster controller
	4. This is where we define the data source connections (links to the databases, it is the alias). It is totally possible for more than 1 corp to share a data source but this is where we set them up and assign them to the cluster controller. This deals with world building and is on level 3 (cluster)
154. state\_tax\_tables
	1. Cluster level
	2. Full id insert
	3. Need all data and id’s (records 1-?????)
	4. This is literally every single state that has stored tax settings. This is for storing payroll settings. These are updated every single year or as things change. It is just a lookup table to help with payroll.
155. statement\_history
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is where we record any history action of statements. As a note statements are any invoices either created or parts still owed by customer. If the customer comes in 20 times and only paid for 8 invoices then the statement will consist of the remaining 12 unpaid invoices.
156. statement\_line\_items
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This where we record the individual invoice information. What they were, what has been paid, age, and status. This is basically the line-items but for your invoices. It is a summary of the invoices.
157. statements
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. The actual statement is where you record who it is, the totals, any notes to the person and due dates. This is the main container that will hold all of the invoice details. This also does contain information on your aging (30, 60, 90, over 90 days etc.)
158. stock\_inventory\_web\_settings
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is a special table to hold ecommerce and web settings for your stock units (big ticket/serialized inventory). This includes things such as special descriptions, alternate messages, any sort of reviews, specs, video lings, and other web presence settings.
159. stock\_photos
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is where you store the references for your jpeg files. Basically your photo gallery for you stock/big ticket items. This deals with what photo number it is, what sort it is, what caption, that type of stuff.
160. store\_line\_items
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is tied to your store locations and is basically special preset values needed for when selling stock units. This is things like special fees, taxes, dealer pak, dealer handling, shipping, postage, documentation fees, title fees, etc. These are all things that are specific to a location for a specific make or model or inventory type.
161. store\_location
	1. Solar system level
	2. Full id insert
	3. Special query – we need two records. They are 1 and 22. 1=Dummy and 22=Void
	4. This is huge, this is where you physically you get to put your location and tax information. You can have unlimited number of locations and tax settings but this is how you stock you model. If you are a mom/pop shop you have 1 location. Think of it like pancakes, how many do you want. Note that the word store or location has corp wide settings to change how you refer to it (store, facility, office, cost, center, job, etc.) This particular table literally connects with every other piece inside of the system. This is how we tie it all together. For instance inventory, deposit, expenses, po’s, balance sheets can be, elements of time, invoices, quotes can all be tied to locations.
162. store\_percentages
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is a way to archive percentages incase you want to split costs or prices between your locations. If you only have 1 location then this doesn’t even play in but if you have multi location you can set your percentage and you can split things like your electricity, insurance, special deliveries, vendor credits, etc per locations. You can archive it because it is date specific to help with when you change the number of locations so you can go backward in time if needed.
163. sub\_inventory\_attributes
	1. solar system level
	2. normal insert
	3. only insert dummy record (id=1)
	4. This this tied to sub inventory. This is the heart of sub inventory. Each attribute is tied to a part category and is specified whether or not it is text, numeric or date attribute. The users get to define these attributes, setup defaults, define requirements and setup labeling options. These sub attributes are available for when they bring in sub items on a PO. This is unlimited and tied to the specific part categories.
164. sub\_inventory\_types
	1. Solar system levels
	2. Full id insert
	3. Need all data and id’s (records 1-????) some are required, some are corp specific.
		1. If it has a corp id = 1, grab it (generic) and ignore all the other corps that are not their own corp.
	4. This is a sub of the stock units. What this does is it is a high level indicator for different types of big ticket items. For instance for a trailer you might have open, utility, dump, stock, etc. Another example might be for vehicles you might have something like vans, SUVs, trucks, autos, etc. This is sub classifications for main inventory times, sort of a secondary level. You have inventory type – then sub inventory type – then makes/models. If you setup your own inventory type, you can actually dynamically set these. For instance, if you are selling guns it might be types of guns or if you are selling modular homes, this would be types of homes, etc.
165. sub\_special\_references
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s
	4. This is a indicator table for the sub\_flags\_and\_tags under elements\_of\_time. This includes things such as tag, flag, progress, category, type, grouping, sub location, phase, etc. These are use to help denote what type of special flag the user is setting up for elements of time. Currently there are 5 different options per time template. As some future plans, we would love to roll these special flags and tags into all other major system players such as invoices, customers, inventory item/parts, POs, vendors, expenses, etc. This will allow us to do some special cataloging.
166. task
	1. solar system level
	2. normal insert
	3. dummy record (id = 1) \* might need to create a dummy record.
	4. This was one of the first implementations of a task queue. Currently this is only being used for reoccurring receipts. It has potential to be expanded but was built as a one-off to help with a specific project. Currently the existing table has a way of adding data and deleting data, but it has no way to update records. \*the underlying code has not been checked. \*\* maybe rename to reoccurring\_expenses since it is specific to that.
167. task\_queue
	1. solar system level
	2. normal insert
	3. dummy record (id = 1)
	4. this is a physical task queue where we can coordinate when to run things. We built this on purpose to be more dynamic so it can tie into other pieces as compared to the task table. It is very similar to the task table but might want to merge the task table into this task queue. As a note we have had many requests for a task queue of sorts and so this was built but may need expanding to meet those various needs.
168. task\_queue\_history
	1. solar system level
	2. normal insert
	3. dummy record (id = 1)
	4. this is a one to many history on if things were successful or not from the task queue. It usually gets a record if the first try fails. The task queue deals with the first attempt and this history table only records on repeats, fails, or roll-over actions.
169. tax\_categories
	1. Solar system levels
	2. Full id insert
	3. Need all data and id’s (records 1-18)
	4. This is a fixed table used for lookups. This includes all kinds of stuff. Two of the most important ones are id 2 = taxable, and 17 = with tax included and have special logic. All other ones equate to a calculation of 0. This table is both used in parts and general inventory and customer assignments. It is also used in my cart favorite buttons and recipe builds.
170. tax\_filing\_status
	1. Solar system levels
	2. Full id insert
	3. Need all data and id’s (records 1-3)
	4. This is for payroll. This deals with what your filing status is, 2 = single or 3 = married. This deals with both federal and state withholdings.
171. testing
	1. cluster level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is a special table to monitor activities for special backend admin pages. If you are doing special uploads or special SQL type actions, this is the table where we are logging what is happening. The name testing is kind of a misleading, but didn’t want to name it back door monitor.
172. third\_party\_error\_logs
	1. solar system level
	2. normal insert
	3. only insert dummy record (id=1)
	4. This is a special error or history log that we can code to. For instance, we can watch who goes to what page (at a page level), who clears a shopping cart, what were the content of the cart, etc. This is tied to permission and the person who is searching it must have some inside knowledge to know how to look at the data. If we need a generic history or log file, this is our guy.
173. tie\_in\_flex\_grid
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This table is one of the most flexable tables in the whole system. It does have to have a host which is one of the 12 main players, but it can be setup with special custom fields, one-to-many relationship building, cross-tying or chaining objects together, or recording any special notes/logs for a single item. To explain all the possibilities for this table would take a whole help file, we can bridge anything using this table. It is like duct-tape for the entire system. If you want to see more info, go to <https://www.adilas.biz/top_secret/help.cfm?id=323&pwd=flex>.
174. tie\_in\_flex\_grid\_history
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This will track any modification after the fact. This only happens if something gets modified. Any initial flex-gird-tie ins get logged in the flex-grid-tie-in table, this is just for modifications after the fact.
175. tie\_in\_flex\_grid\_titles
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is a special table that allows you to name the custom fields. As a note, all 12 main player types may have up to 15 custom fields per type. Some of the different types of titles allow for open text entry, prebuilt drop downs, or combo text/drop down options. This table plays with the tie\_in\_flex\_grid table as well as the tie\_in\_title\_subs table. For more information go to : <https://www.adilas.biz/top_secret/help.cfm?id=324&pwd=flex>.
176. tie\_in\_title\_subs
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is where you get to build the drop downs for your tie in flex grid types. For example, say you wanted the colors red, blue and green, that would take up 3 entries in this table but would tie into 1 tie in flex grid title or field name. the flex grid allows you to make your own custom database, this is where you can store some of the custom lookup values.
177. tie\_in\_title\_types
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-3)
	4. This is how we denote if it is an 1 = open entry field, 2 = drop down, or 3 =both. This is part of the flex grid titles or custom field name options.
178. time\_action\_logs
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This a sub for elements of time. This section can be turned on or off and allows for quick flipping of pre-set actions. Some of the actions deal with starting, stopping, pending, completed, etc. As a note, these actions do not allow any notes and they do quick math to help keep track of when things flipped action statuses. As a note, for the future we would love to actually allow the users to build in their own time action statuses.
179. time\_actions
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-13)
	4. These are the preset time actions that go into the time action logs. These are things like started, stopped, pauses, in process, pending, completed, check-in, checkout, etc.
180. time\_card\_history
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. Time cards actually get a history if someone modifies it. The time cards are where people clock in and out using the system, like an employee. These history things happen if a manager or admin go in and modify something and/or lock it.
181. time\_cards
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is the actual table the holds the clock in and out date times for employee users. These values are automatically pulled into the payroll calculations. There are 3 different levels of access, admin, manager, and employee/user. These different levels are all permissioned. All time cards are tied to departments and users.
182. time\_frequency\_types
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-)
	4. This is for elements of time. These are things like seconds, minutes, hours, days, weeks, months, years, etc. These are used for budgets inside elements of time. For example, if you wanted to predict how long something will take? 3 hours or 4 days, etc. it was setup as standard units of time. As a note, it is prepping for rentals and reservations.
183. time\_photos
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is the photo gallery table for elements of time. It only allows jpeg’s but allows up to 100 photos for each element of time or event. This is great for small work session, picture of the white board, screen shots, or pictures of a trip to the zoo for your class. Photos may include captions, dates, sort numbers, etc. the system does all the sizing and categorizing behind the scenes.
184. time\_sub\_assignments
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is a sub of elements of time. This was originally setup to be the “any assignment” piece. This was going to be like flex-grid but specific for elements of time. The reason why we called this the any assignments is because we want it to handle any person place or thing and how they interact with other system players. This could be as simple as certain clients and users meeting of ran appointment, or as crazy as full dispatch where you have full on workers, equipment and supplies. This is missing a couple pieces and still needs to be built out. For more information go to <https://www.adilas.biz/top_secret/help.cfm?id=391&pwd=sub>.
185. time\_sub\_comments
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is a sub of elements of time and allows for unlimited notes/comments and even forum type chaining of different entries. This allows you to add all kinds of details, additional descriptions and track a virtual timeline or conversation in a single element of time.
186. time\_sub\_dates
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is a sub of elements of time and it is specific to catch date times and amount of times that pass, either between pieces and stages or as a timer. This is used for recording actions that take place. Users can clock in and track time or just general layout of expectations and proposed phases/timelines.
187. time\_sub\_gps
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is a sub of elements of time and is specific for tracking things that need coordinates. This includes an x, y and z level or axis. This could be physical GPS stuff, RFID tags, exact printing specifications, etc.
188. time\_sub\_inventory\_[corp\_id]
	1. solar system level
	2. normal insert
	3. currently no dummy record. This has got to be corp specific table.
	4. This table is tied to sub inventory for parts and general inventory items. This is considered packaging, lots, batches, etc. Basically it is fed through PO and invoice\_line\_items to track sub inventory levels of detail. That could be special costing, pricing, quantities, RFID tags, etc. This is considered the child inventory table. It has ties to both PO invoice line items and parts and vendor payees. As a note, for future build out we would love to tie this eventually to elements of time to help track more historical data.
189. time\_sub\_inventory\_history
	1. solar system
	2. normal insert
	3. only insert dummy record (id = 1)
	4. this will eventually hold additional information on who is modifying and editing sub inventory. It wont do anything on the initial creation but will be on modifying after the fact. This is currently not active, and has not been tied in yet.
190. time\_sub\_payroll
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is a sub of elements of time. This will deal with sub payroll options to help with the payroll process. Currently this table has been created but has not been linked or wired up. The goal is to actually create sort of a timesheet for element of time. Current payroll currently pulls form stock units, invoices, deposits, user time card, etc. This would be another way to add independent payroll pieces to the payroll section. As a note we have some clients that could really use this feature. This needs to be built in if we really want to take payroll to the next level.
191. time\_sub\_pool\_assignments
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is a sub of elements of time and actually is kind of like players that are part of the time\_sub\_pools are used and pulled into the time\_sub\_assignments. Basically it is how we actually make those connection. Some of this still needs to be worked out and was added but no one has really detailed this out yet or tied it in. This is how we add people and or things to the time\_sub\_pools. We then use those time\_sub\_pools to add them to the time\_sub\_assignments.
192. time\_sub\_pools
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is a sub for elements of time and is an optional piece. This is how you create small specialty groups, common groups, or commonly used pieces. We foresee pools being used for pools of users, technicians, parts, assets, equipment, stock numbers, locations, rental fleet units, vehicles, etc.
193. time\_sub\_reminders
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is a sub of elements of time and is a way for us to communicate back to the user or manager as to what needs to happen. This is things such as warnings, reminders, notifyers, or put some sort of a task in front of you. As a note, this may be automated by the system or it could be pre-set to send the notification at a certain time. This table also allows setup for whether it is text or email and who the recipients are. Currently this is not wired up. An example could be to remind manager when part quantity gets to 10. As a note, this will probably end up using the task queue.
194. time\_sub\_sign\_offs
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This a sub of elements of time. This allows for one or more persons to sign off on different tasks or aspects for the current event or element of time. This allows for an unlimited number of people to add their 2 cents (notes) and virtually sign off on work done, progress made, things discussed, agreements made, etc.
195. time\_sub\_special\_flags
	1. solar system level
	2. normal inster
	3. only insert dummy record (id=1)
	4. this is a sub of elements of time. This is where we hold all the sub flags, tags, phases, sub locations, progress reports, categories, and types. This particular table and feature combines sub dates and times, sub action statuses and has its own flagging and tagging capacity (huge potential). This is a giant which is laying dormant right now.
196. time\_templates
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is even above an element of time. This is how you setup and name a template. A template act like a cookie cutter and the elements of time are the cookies they produce. This table holds a number of toggle on or off settings to help the element of time know how to function and what the default values are. For more information go to <https://www.adilas.biz/top_secret/help.cfm?id=383&pwd=template>.
197. time\_zones
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-8ish)
	4. These are the general time zones that are tied to the employee users. This table also includes things like the timezone offset which helps us match up datetime stamps per user with where they are hitting the server. Each server is setup to their time and the offset is to change the server time to match with the users local timezone. This is important because every history record eventually taps into these timezone offsets. Also as a note, when we go through daylight savings time these need to be manually updated on all servers.
198. title\_account\_history
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This deals with floor plan. If you have big ticket items you have a title. This is part of the title\_accounts table. This looks for what was the title and what was it’s value. Typically this is flipping from bank to bank, to in-hand, pending, sold, etc.
199. title\_account\_types
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-3)
	4. This is a small lookup. 1 = dummy, 2= holding account (like in-hand, sold, pending, etc), 3=finance account (specific bank entities or title account companies like GE).
200. title\_accounts
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is corp specific and how you create and name. This is where you name your external banking floorplan or internal holding accounts. This is how you organize your titles. By default every corp gets some of these by default (hard coded in maintenance.cfc as the corp gets built).
201. transition\_invoices
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is basically in between a quote and invoice. Also known as a QTI = quote to invoice. If in this state it flags it and keeps a history for how long it was in this transitional state. It has a start-date and end-date and tracks it differently from financials. This is like layaway (counts against inventory, but not as revenue). This is kind of like an IOU from the customer. In this state, it doesn’t show up in revenue, taxes, accounts payable. You can also accept a deposit on the merchandise. All of these transactions are automatically shown on the balance sheet until it leaves this transitional state.
202. units\_of\_measurement
	1. Solar system level
	2. Full id insert
	3. Special query – we need all records that have a corp id of 1.
	4. This is actually how we determine how things are bought and sold (example inches, feet, ouces, pounds, tons, etc.). Custom measurements can also be created by the user. All inventory is tracked based on the lowest common denominator. Currently we can kind of do conversions, but that is being built out, as well as custom convertions.
203. usage\_classes
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-unknown). Grab everything with a corp\_id = 1 (general), everything is specific to that corp.
	4. This deals with stock units. This is meant to be used for searching but is not currently being used to it full potential. This is meant to be a type of dynamic search for stock inventory (like flags and tags for stock units and ecommerce). Basically this is a multi search criteria.
204. web\_page\_api\_assignments
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is a sub of the web\_pages table. This has a couple different usages. 1. It is used internally when we assign an Adilas API socket to a corporation. This is how we virtually open and close doors within the system. This is also how we control all 3rd party integration within the same table.
205. web\_page\_api\_history
	1. Cluster level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is a sub of the web\_pages table. This is literally who accesses what. This is done externally. This tracks who access what method, what IP address, what time, etc. This is kind of like the surveillance camera for the API connections.
206. web\_page\_comments
	1. Cluster level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is a sub of the web\_pages table. These are page level comments that get recorded for .cfm and .cfc files that get logged under the web\_pages table. The original goal was to make all pages free and make money off of the services and not the code. If this was all documented then this could be searched and allow others to see just the comment portion without showing the real code. This is to help make it cool for search engines or allow someone to look through comments to know what was going on in the page without seeing raw code.
207. web\_page\_history
	1. Cluster level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is a sub of the web\_pages table. Primarily this is the documentation on who is creating/modifying stuff. This is who is touching what, who is changing the documentation.
208. web\_page\_modified\_log
	1. Cluster level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is a sub of the web\_pages table. The modified logs are when the different developers access or make changes to the files. This is kind of a running history of who made changes at a high level. This is a user maintained history per page/function.
209. web\_page\_parameters
	1. Cluster level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is a sub of the web\_pages table. This table was setup so we could document any inbound pieces that could be form scope, variables, URL, arguments, etc (what is coming in and what are their defaults). Currently this is very heavily used for documenting public Adilas API sockets (method arguments).
210. web\_page\_results
	1. Cluster level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is a sub of the web\_pages table. This is actually what the page produces. This is the outbound on whether it is html, JSON, web, other. For instance what does it return (JSON, structure, query, string, etc).
211. web\_page\_samples
	1. Cluster level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is a sub of the web\_pages table. This is a storage container where we can put JSON, wddx, XML and other API samples. This could be for calls or samples on what is being received.
212. web\_page\_settings
	1. Solar system level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is a sub of the web\_pages table. This table is tied to the web pages but will hold all sorts of JSON or extra storage. Currently this is the generic JSON storage for the whole system. This could be page settings, report settings, 3rd party settings, favorites, etc. If we need to store JSON, it is probably in here. This storage can be done per corp, per individual per corp, or setup as a one-to-many per person or corp.
213. web\_page\_sub\_results
	1. Cluster level
	2. Normal insert
	3. Only insert dummy record (id=1)
	4. This is a sub of the web\_pages and web\_pages\_results tables. This table stores sub details from different transactions. For example if we call a function and we get an ok success message we store all the other information passed back (ex: query, struct, array, variables, etc). This is a sub of the results.
214. web\_pages
	1. Cluster level
	2. Full id insert – built on data 0 and pulled across using the developer homepage update tables section, no matter what.
	3. Pull them all over from data 0 (currently 700+)
	4. This is a special table that helps us get id numbers for things that are going to be accessed externally. This table currently is setup to handle pages (.cfm) pages, full components (.cfc), internal functions and API methods, 3rd party products, and even custom page settings. This section is built on a giant one-to-many (the master table for pages, functions, and external APIs). This table connects to all other tables that have “web\_page” as part of their table name (10+).
215. web­\_part\_options
	1. solar system level
	2. normal insert
	3. only insert dummy record (id=1)
	4. This is a one-to-many that is a possible add on for the parts and general inventory items and stock units. The main goal of this table is to store extra settings that are needed to record data for ecommerce. These are things such as specs, reviews, video links, key works, usage, and other search options. This table is able to handle and hold HTML based code that is used on eCommerce and web based presence.
216. web\_price\_settings
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-6)
	4. This deals with eCommerce. This is set on the single item or stock unit item level. It is a setting that tells the web what price to show. These are things like run at current price, special web pricing, hide/don’t show, or 0.
217. work\_shift
	1. Solar system level
	2. Full id insert
	3. Need all data and id’s (records 1-4)
	4. This deals with payee records and which shift they are associated with. This is a simple lookup for things like day, swing, grave, etc. This is primarily a payroll and time card feature. The only option that has special code is grave, where when selected it will automatically advance their time card.
218. zipcodes
	1. Cluster level
	2. Full id insert
	3. Need all data and id’s (records 1-70,000+)
	4. This is literally just a list of all zip codes to help with lookup the zip code by city and state. It is basically a zip code lookup helper.

Step 4: We need to add some additional formatting for all dates, date/time stamps, and decimals. They are:

1. All dates need this code: #CreateODBCDate(qryGetDummyRecord.)#
2. All date/time stamps need this code: #CreateODBCDateTime(qryGetDummyRecord.)#
3. All decimal numbers need this code: #Trim(NumberFormat(qryGetDummyRecord., 9999.99))# or #Trim(NumberFormat(qryGetDummyRecord., 9999.99999))#
	1. as a note on decimals, the number decimals need to match up. For example, 2 decimals needs a mask of “9999.99” and 5 decimals needs a mask of “9999.99999” as needed. The 9’s are basically a placeholder value for a thing called a mask. It doesn’t really mean 9’s.

Step 5: We need to figure out which tables will be part of the solar system and which tables will be on the cluster (box or domain) level. We then need to make new master database files. These new files should be ultra clean (both cluster and solar system levels). These will become the base that we will build on. Take the time to get it right.

1. Prep it locally with dummy line items.
2. Post the database on a live server to get the actual values.
3. After getting live look-up values. Bring the new databases back down for local testing and storage.
4. Save a copy and archive it as the new master copy. It should be fairly small in size but super clean.
5. Delete old tables that are not in use and there are no plans for them. This may include a search of the code for specific tables and how they are used.
6. Create webservice feeds (API connections) for cluster specific tables such as permissions, web pages (documentation), help files, and tax tables.
7. As needed, alter table names, and it is ok to put tables on both levels if needed. This is permission to do what needs to be done… Dream it up!

Step 6: Get things ready for copying from the old master to the new mini versions. This will involve queries on each table and then inserts and updates depending on if the value is in the database or not. This may take some extra time, allowing it to go both ways (insert or update) based on record count. We will also need to add pagination to this routine as some of the tables have thousands and thousands of records. This is the copy and migration part of the puzzle.