**I. Building Applications with Force.com – Part 1**

**1. Introduction to Force.com**

* Salesforce delivers applications to manage sales & customer service operations.
* These applications include:
  + Sales Cloud – managing sales operations
  + Service Cloud – customer service
  + Chatter – for collaboration
* Force.com is platform underlying Sales Cloud, Service Cloud, and Chatter app.; is platform as a service (PaaS); for build, configure & customize app. in cloud.
* Force.com provides 3 tools to migrate metadata based on Metadata API: Change Sets, Force.com Migration Tool, and Force.com IDE.
* Setup menu – declarative interface is a point-and-click tool

**2. Designing Applications**

* Before design, answer basic design questions:
  + Who are your stakeholders and business partners?
  + What are the business requirements?
  + Who will use the application?
  + What reports do you want to generate from the application?
  + How will people learn to use the application?
* Additional considerations of Application Design:
  + Keep the design simple: do not add complexity where it doesn’t add value; include only processes that benefit your organization
  + Validate design with a user group: test whether or not your app. design will work the way it is intended; implement the feedback before you deploy app.

**3. Building Your Data Model**

* Standard fields:
  + ID: is indexed, 15 char case sensitive, 18 char, case insensitive, first 3 chars consist of code that identify type of object
  + Name: is indexed, required field, text (need not be unique)/auto number (usually unique), appear as first column by default in list/related views
  + Owner: can represent group/user, has additional privileges, default owner is creator
  + Other system fields: created by; last modified by; created date; last modified date
* Custom fields: include some standard fields
* When delete, it is stored for another 15 days after deletion or until org. deleted them
* Limit on custom fields depend on Salesforce edition
* Fields can be made universally required: Currency, Date, Date/Time, Email, Master-Detail Relationship (always required), Number, Percent, Phone, Text, Text Area
* Data types:
  + Numeric – number, currency, percent
  + Calendar – Date, DateTime
  + Limited Option – checkbox (true/false value), picklist, picklist (multi-select)
  + Formatted Text – Email, Phone, URL
  + Text – Text (255 – single line), Text Area (255 - multiple), Text Area (Long – 32768 - multiple), Text Area (Rich - 32768, images, links, formatted), Encrypted – any amount
  + Calculation – Auto Number (system generated), Formula (read-only), Roll-up summary (created on master object, read-only, display sum, minimum, maximum value of a field in a related list or record count of all records listed in a related list)
* Properties
  + Name – used programmatically
  + Label – used in UI
  + Universally required field – required field for all record types and will always display on the edit page
  + Unique field – cannot contain duplicate values, can be case sensitive/insensitive
  + External ID – record id from another system. Fields of type: Text, Number, Email. Can have up to 3. Will have custom index, increases report & SOQL performance
  + Default Value – can be based on formula
  + Help Text – field-level info in the form of hover text
* Standard picklist fields can be controlling fields, but not dependent fields, max # allowed in controlling field is 300. In addition, if a field represents both a controlling field and dependent field, it cannot contain more than 300 values
* A custom multi-select picklist cannot be the controlling field for a dependent field
* Encrypted fields
  + The value of the encrypted field is visible to users who has ‘View Encrypted Data’ permission
  + Cannot have default values and cannot be unique or external ID
  + The encrypted field can be edited irrespective of the ‘View Encrypted Data’ permission
  + Fields are encrypted with 128-bit keys and use the AES algorithm. They are 175 characters in length.
  + Some limitations on the use of encrypted fields:
    - Encrypted fields are not indexed and hence not searchable.
    - They are not available in filters.
    - Encrypted fields cannot be used in where condition in SOQL queries.
    - In an e-mail template encrypted value is always masked, irrespective of ‘View All Data’ permission.
    - To work with encrypted custom fields, we need to make sure the organization has secure connections using SSL.
    - If you clone a record that has encrypted custom fields, Salesforce will copy the data from the field only if the user has the ‘View Encrypted Data’ permission.
* Lookup Relationship (1:n)
  + links custom or standard, independent ownership, security, each object can have up to max of 25 lookup relationships, made required/optional, for optional,
  + if field is optional, can use one of the following actions when a lookup record is deleted:
    - clear value of this field (default)
    - don’t allow deletion that is part of lookup relationship
    - delete this records also.: cascade delete bypasses security & sharing settings, allowing users to delete records they don’t have access to.
* master-detail relationship (1:n)
  + tightly coupled
  + parent field is always required
  + automatically deletes the child when parent is deleted, ownership & sharing are inherited from the parent to the child.
  + cannot have sharing rules cannot be used in manual sharing or queues
  + cannot contain a standard object on the detail side of the relationship
  + master-detail can be configured so that child records on the custom objects can be reparented
  + cannot create master-detail relationship to users & lead objects
  + page layout should include a field in the child object for the associated master record
  + Up to 4 nested levels of Master-Detail relationships can be created, counting from the master object to the most deeply nested child object.
* Difference between Lookup & Master-Detail Relationship
  + Lookup
    - max 25 objects
    - parent required or optional
    - security & access are independent for objects
    - deleting parent object may delete the child, if the child field is required
    - link objects across multiple layers
    - lookup field on page layout depends on required/optional choice
    - cross-object field updates and roll-up summary fields cannot be done
  + Master-Detail
    - max 2 objects
    - parent field on child required
    - access to parent determines access to children
    - deleting a parent automatically deletes the child
    - link objects across limited no. of layers
    - lookup field on page layouts is required
    - cross-object field updates and roll-up summary fields can be done
* Reporting implications:
  + Lookup relationship joins data from 2 related objects
  + Master-detail relationship joins data from 3 objects: master, detail, one other lookup object
  + A primary or master obj. determines the visibility of records
  + Reports can only display one child obj. at a time
* Special relationships
  + self-relationship (lookup) & many to many relationships (master detail relationship)
  + junction object
    - custom object with 2 relationships
    - also referred as intersection object
    - moved to recycle bin when any of the associated master records are deleted, however deleted permanently and cannot be restored when both associated master records are deleted
* Lookup Filters
  + to limit the search results for related fields (lookup, master-detail or hierarchical)
  + filters can be required or optional, value must match the criteria, if it is required

**4. Building Your User Interface**

* Cannot create tabs for standard objects. The Standard Object Tabs can be hidden/ shown and added to any custom application
* 3 different ways to customize UI: custom apps, custom tabs, custom layouts
* Custom apps.:
  + Consists of all objs, tabs, processes, and services associated with a given business function.
  + Consists of a name, desc., ordered list of tabs, (optionally) custom logo & landing page.
  + File size for custom logo is 20 kb, GIF or JPG, 300 wide by 55 pixels high, add file to documents tab to have it as a header
* A custom tab:
  + is a user interface component you create to display custom object data or other web content embedded in the application
  + 3 types of custom tabs (custom tab has a unique tab style)
    - custom object tabs – display data from any custom object, provide the options to create, read, update, and delete an object.
    - web tabs – display any external web app in UI tab
    - visualforce tabs – display vf or html page in UI tab
* Page Layouts
  + Defines the org. of fields & custom links for custom app.
  + Customizations include field locations, page section customizations, field properties by setting them as visible, read only, required.

**5. Introduction to Business Logic**

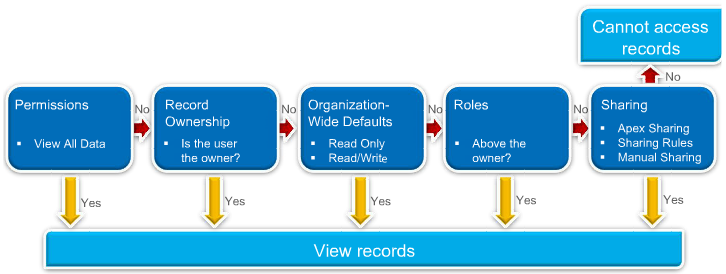
* Custom formula fields
  + Are smart custom fields, can be used to build business-specific calculations using simple wizards & Excel-like formulas.
  + Can reference standard, custom, other formula fields, or fields on related objs.
* Cross object formula
  + can be created from objects that are linked either by a master-detail or a lookup relationship
  + can contain objects that span across multiple levels of relationships
  + limit: 10 unique relationships per object across all formulas and rules
  + exceptions:
    - cannot reference in rollup summary fields
    - cannot reference merge fields for objects related to activities
    - cannot reference record owner merge fields for any object
    - If a standard and custom field have identical names or labels, the merge field displays the custom field value.
    - If two custom fields have identical names or labels, the merge field may display an unexpected value.
    - If you create a field label called Email and a standard field labeled Email already exists, the merge field may be unable to distinguish between the fields. Adding a character to the custom field name makes it unique. For example, Email2.)
* Roll-up summary fields
  + Calculate values from a set of related records.
  + read-only formula fields that can display sum, min, max or record count
  + can add for all custom master-detail relationships & some standard relationships
  + can create roll-up summary fields that automatically display a value on a master record based on the values of records in a detail record. These detail records must be directly related to the master through a master-detail relationship

**6. Migrating Configuration Changes**

* Configuration changes are stored as metadata
  + (new sandboxes that are not activated within 30 days & the sandboxes that have been locked for 30 days will be deleted)
* 3 deployment tools to move metadata: change sets (point & click interface), force.com IDE (development environment), force.com migration took (ANT based)
* Change set:
  + Is a point-and-click Web interface
  + Move only metadata, not data
  + Have a functionality like email, you send an outbound change set to org, admin on destination org receive inbound change set to review or deploy
  + apex code should meet 75% of covering unit tests
  + system detects incompatibilities among versions
  + components are not available in a change set need to be migrate manually
  + ability to work with change sets is controlled by profile permissions
  + change sets cannot be modified once it is uploaded but can clone and modify a clone.

**7. Controlling access to records**

* Record ownership
  + has an owner, sharing based on owner of a record, ownership can be transferred to any user who at least has read access
  + child records in master detail relationship do not have owner, they inherit ownership from parent record
* Types of owners
  + users – full access if a user is owner. if read permission revoked, then they can’t see their own record
  + queues – allows multiple users to take ownership, assigned manually or thru assignment rules
* Record Access
  + User wants to view a record, system check: profile to verify object level access; validates sharing model of the org. -> determines if user can view or edit; determines which user has access, what level of access, how they acquired that access.
  + read only access, read-write access, full access (delete/share/transfer)
* Ways to obtain record access
  + Full access: owner, above the owner in role hierarchy, contains ‘modify all data’ permission in profile
  + Read/Write or Read only: organization-wide defaults, role hierarchy, sharing rules, manual sharing, apex sharing, ‘view all data’ permission
* Profiles vs. Sharing Models
  + profiles
    - control access to objects and fields
    - control whether user can view positions that user can view
    - which fields the user can view & edit
  + Sharing models
    - control access to records
    - control the positions to view
* OWD
  + security settings that define the base line level of access to records that the user doesn’t own
  + only way to restricts access to data in sharing model
  + 3 level of settings
    - public read-write
    - public read only
    - private
  + Determining OWD
    - identify the most restricted user of this object
    - will there be an instance of object that this user is not allowed to view
    - if yes, then owd is **private**
    - else,
      * will there be an instance of object that this user is not allowed to edit
      * if yes, then owd is **public read only**
      * else, then owd is **public read-write**
  + setting owd for child records in master-detail relationship – child inherits  owd from parents
  + child records in lookup will not inherit owd
  + it is possible to change owd any time, but it may have consequences
  + owd can be set for both standard & custom objects
* Roles
  + control the level of visibility to org data
  + every user associated to role
  + assuming no sharing rules created, users in the same role cannot access each other’s records
* Role Hierarchy
  + defines data access rights granted to users at higher roles
  + users access to all records they own and their sub-ordinates
* record access rolls up with role hierarchy with all standard objects
* with custom objects, a setting named ‘Grant Access using Role Hierarchy’, this can be prevented
* Public Groups
  + Roles are two dimensional structures. Public groups are way of grouping users together to grant them record access.
  + Groups are good way to extend access across the nodes in hierarchy tree
  + ‘All Internal users’ is a default public group. Public groups can be made up of any combination of users, roles and subordinates and other public groups.
  + Can use public groups in a sharing rule to reduce the number of sharing rules
  + Public groups can also be used for folder access.
* Sharing rules
  + are created to grant access to records between users when access does not roll up
  + Using sharing rules, read only and read/write access can be granted to users
  + Sharing rules cannot be more restrictive than owd settings.
* Manual Sharing
  + used to grant access to records on a one-off basis when random users require record access.
  + Access rights can be granted by the owner of a record, anyone above the owner in the role hierarchy and by the system administrator
  + It is granted at the record level and is not used to grant access at the organization level
* Apex sharing reasons
  + allow developers to define the reason why a user or group of users have access to record
  + apex sharing reasons exist only for custom objects and they are defined for individual objects.
  + each object can have up to 10 apex sharing reasons
  + sharing rule has to be created manually using new manual sharing rules
  + deleting apex sharing reasons will delete all manual sharing rules associated with it
  + Users with ‘Modify all data’ permission can change sharing using apex sharing reasons
  + apex sharing reasons should be used programmatically and not through the application
* Controlling Access to Data Review



**8. Design Considerations**

* Design security of app, need to identify users & their access path
* Users: need to know the users or actors of the app
* Data: need to identify
  + data that can be accessed by users
  + data restrictions and revoke access from such sensitive data
  + users who should be allowed to customize the app and administer app.

**9. Managing your user’s experience**

* Types of licenses
  + every user must have a user license
    - defines the functionality permitted for the user and determines the profile available to the user
    - can have only type of user license, but may have many feature licenses
  + Two types of license
    - Salesforce
      * full access to standard CRM, force.com custom, AppExchange apps, standard or custom apps
      * users with this license are entitled to access any standard or custom apps
    - Salesforce Platform
      * Access to custom apps, force.com custom, AppExchange apps, no access to CRM functionality
      * Can use core platform functionality such as accounts, leads, contacts, reports, dashboards and documents
  + can have add-on features licenses for specific features, such as apex mobile user, sf crm content user, marketing user
  + can have more than one type of feature license
* Profiles
  + define user permissions to perform different functions
  + every user has a profile that defines what user can do, how user can view records that they have access to
  + each profile is associated with a license type
  + profile work with sharing models or role hierarchy
* Functions of profiles: profiles control permissions, access to data and the UI
  + Permissions
    - Control administrative & general user permissions
    - define all actions that a user in a profile can perform
  + access to data
    - controlled by field level security settings which allow admin to define object and field level permissions
  + user interface
    - page layouts, tabs, applications available for each user, record types for each profile determine what the users will see when they login
  + two types of profiles
    - standard
      * can’t be created or deleted and permissions cannot be customized
    - custom
      * create or clone the standard profile and modify the settings
  + types of permissions
    - Administrative
      * can grant some administrative permissions to custom profile
      * View All Data – allows administrators to view all records regardless of all other security settings
      * Modify all Data – allows administrators to modify all records
      * Customize Application – permits administrators to administer application
      * API only user – cannot login to sf.com. such users can only use the application through API calls
      * Password Never Expires – prevents password expiring
    - General User
      * control the ability that standard user can do like editing tasks
    - Standard Object
      * control read, create, edit and delete action on standard objects
    - Custom Object
      * control read, create, edit and delete actions on custom objects
  + Permission Sets
    - collection of settings and permissions that provide users with access to various tools & functions
    - allow to grant permissions to specific users in addition to their existing profile permissions
    - represent a concept like job title
    - handle the system requirements that previously existed on the profiles
  + user can have only one profile, but can have multiple permission sets
  + while assigning permission to users, use profile to assign most restrictive settings and assign additional permissions using permission sets
  + an org can have up to 1000 permission sets
  + permission sets can be used assign additional permissions, but not to deny permissions
  + permissions can be revoked by
    - removing permission from permission set
    - changing a user’s profile
    - disabling a permission set in a profile
  + use permission sets to grant permissions for
    - applications, objects, fields, tabs, record types, apex classes, service provider, visual force pages
  + permissions that are not available in permission sets must be set through profiles (login hours, ip access, etc)
  + revoking delete permission for a child object in master child relationship will not prevent deleting the child record if parent record is deleted
* Field-Level Security
  + restricts access to fields
    - list views, reports, force.com, connect offline, custom links, mail merge, related lists
  + overrides less restrictive page layout settings
  + set at profile level
  + each profile can have different level of access to object
  + doesn’t allow conditional security of records
  + all users can edit any accessible fields in any record
* Customizing UI and Profiles
  + Record types
    - Users in different profiles may need to view different details on a page.
    - define the manner in which data is displayed according to business needs
    - determine the page layout and limit the picklist options based on the profile & permission set
    - not security tools – do not subclass or partition the data, they work at the UI level and not at the data level
  + Assign record type:
    - New record: set record types of any new record. When assigned multiple, can select an appropriate record type from a picklist
    - Assign record types to a profile: user profiles or permission set determine record types available in the picklist. Add to profile by editing profile
    - Existing record: users can change the record type of an existing record and it does not affect the values in the record
  + Page Layout: help organize details & edit pages.
    - Control fields, related lists, custom links, page sections that user can view
    - User can set field properties through page layout settings & create unique page layouts for different business scenarios.
  + Record types allow developers to specify different page layouts for the same obj so that users in diff profiles can have diff views of same page.

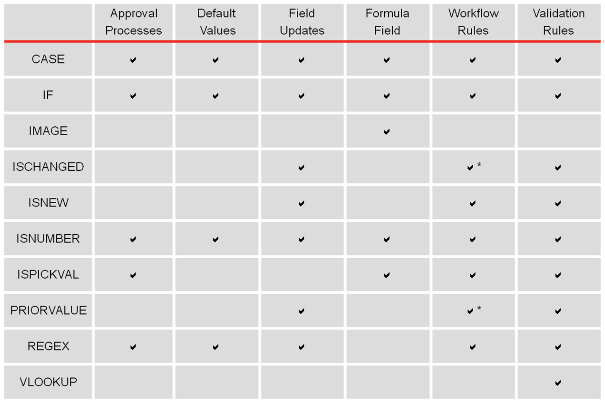
**II. Building Applications with Force.com – Part 2**

**1. Designing Data Access Security**

* Establishing Data Access
  + When you want to determine data access for a object,
    - consider the OWD default
    - the owner of the records
    - uses who need access
    - rules governing data access
  + When determining access to sensitive data, you need to analyze the access requirements and restrictions for each profile

## 2. Implementing Business Processes

* Typical business requirements can be used for
  + preserving data quality
  + automatic processes
  + keeping processes from getting ‘stuck’
  + keeping systems in sync
  + auditing
* Features
  + Formula fields
  + Validation rules
  + Approval process
  + Workflow Rules
  + Outbound Messaging
  + Field History Tracking
  + Setup Audit Trail
* Functions
  + ISCHANGED – compares with previous value and returns true if it is changed
  + PRIORVALUE – returns the previous value of the field
  + ISNEW – checks if a formula is running during creation of new record and returns true if it is
  + ISPICKVAL – determines if the value of picklist is equal to specified string
  + REGEX – string used to describe the format of the string according to certain syntax rules. It compares a text field to regular expression and returns true, if there is a match
  + VLOOKUP – returns value by looking up a record value in a custom object. It checks against a key and returns value from that key.
  + ISNUMBER – returns true if a text value is number
  + CASE – checks against a series of values
  + IMAGE – inserts an image
  + HTMLENCODE – encodes text stings and merge field values for use in html (e.g. ‘<’)
  + JSENCODE – encodes text strings and merge field values for use in javascript (e.g. apostrophe)
  + JSINHTMLENCODE – encodes text strings and merge field values for use in javascript within html tags
  + URLENCODE – encodes text strings and merge field values for use in URLs



* System Logs
  + display logging info, cumulative limits and source code of transaction
  + used for debugging code snippets
  + used to view debug log or execute anonymous code blocks
  + display system resource info
* Log levels
  + from lowest to highest
  + Error – lowest, produces distinct results and only error messages
  + warn – warn and error
  + info – info, warn and error
  + debug – includes low level and calls to system.debug
  + Fine/Finer – system.debug, dml, soql/sosl, entrance and exit
  + Finest – includes all messages in previous levels and on apex scripts
* Debug Logs
  + can record database operations, system processes, and errors that occur when executing a transaction or running unit tests.
  + contain info on database changes, automated workflow processes, validation rules
  + request-response xml, apex script errors, and resources used by an apex script
  + records errors and system processes that occur in an org
  + can be retained and managed for specific users
  + 20 logs can be retained for an org, when max is reached, oldest one is overwritten
  + debug log is different system log
  + system log refers to console link at the top of the page
  + underlying logging system is same
  + system log is live console, debug log is persistent store

## 3. Preserving Data Quality

* Validation Rules
  + used to verify that the data entered meets the standards before the user saves the record.
  + Can contain formulas or expressions that evaluate the data in one or more fields
  + return true or false
  + run on the server after user clicks Save button in UI or data is added through API or Import wizards
  + are executed for fields that are stored in the object, but not part of the displayed page layout
* can be used for
  + enforce conditionally required fields
  + enforce required data formats
  + enforce data consistency (can combine with VLOOKUP)
  + prevent data loss
  + enforce proper data formats
* can be used in conjunction with a roll-up summary field can be used to prevent users from adding or deleting records

## 4. Automating Business Processes with Workflow

* Workflow Rules
  + Enable to automate business processes. When record meets criteria for a rule, an action is triggered
  + Entry Criteria then Immediate Actions or Time dependent actions
  + Steps
    - Specify the object (both standard & custom objects are ok)
    - Select Evaluation Criteria
      * only when a record is created
      * when it’s created or edited and now meets the criteria
      * every single time the record is created or updated
    - Define rule criteria
      * Using filters or a formula
  + Workflow Actions: immediate or later time
    - Tasks – can be assigned to user, role or record owner
    - Email Alerts – can send email to one or more recipients (from address can be current user address or org wide address)
    - Field Updates – can update a field value on a record (including record type/owner)
    - Outbound Messages – can send specific info to designated endpoint in form of API/SOAP message
* Time-Dependent Workflow
  + triggered depending on elapsed time (evaluated off of any date field in Salesforce)
  + time-dependent actions have a time trigger
  + the action is queued to fire
  + Some considerations
    - cannot use time-dependent workflow when a rule is set for evaluation, every time a record is created or updated
    - when a new workflow rule is created, it does not affect existing records
    - can monitor and remove pending actions by viewing the time-dependent workflow queue
    - if a record that has an action pending against it in the time-based workflow queue is modified so that the record no longer meets the criteria or the timing changes, the action will be updated in the queue
    - if a record no longer meets the time-based workflow rule criteria, the action is removed from queue

## 5. Automating business processes with Approval Processes

* automates routing of records for approval
* contain one or more steps and can be logically split into 6 steps
* not automatically sent for approval, user has to submit
* steps
  + process definition
    - it is determined which records should enter the process and what settings should apply to the whole process.
  + initial submission actions
    - developers decide what happens to a record after it is submitted for approval – actions are locking a record, assigning a task, sending an email, updating a field, sending an outbound message
  + step definition
    - developers determine whether all records should enter the step or whether records only meeting the criteria are chosen.
      * if later is chosen, developer defines the criteria for entry to the step. developer also assigns the approver and determine whether the approver can delegate
      * if there are multiple steps, developers can decide what should happen if a record is rejected at a step after the first step: should it go back one step, or should it be considered a final rejection
  + final rejection actions
    - developers define the actions to be taken when a record is rejected
    - actions are: unlock a record, assign a task, send an email, update a field, send an outbound message
  + final approval actions
    - developers define actions to be taken when a record is approved
    - actions are: unlock a record, assign a task, send an email, update a field, send an outbound message
  + recall actions
    - developers define actions to be taken when a record is recalled from the process.
* Workflow Rule vs Approval Process
  + Workflow rule
    - are triggered upon save
    - consist of one set of criteria and actions
    - can be modified or deleted
  + Approval process
    - triggered only when a user clicks submit for approval
    - consist of multiple steps, have entry criteria, step criteria and step actions; have initial submission actions, rejection and approval actions and actions for each step
    - some attributes can’t be modified, processes must be deactivated before they can be deleted
* Skipping steps
  + allows developers to skip steps within an approval process based on specific criteria
  + skip step is a step that has criteria defined to determine whether or not approval is required
  + 3 options
    - go to next step
    - approve record
    - reject record
  + considerations
    - Go to next step” option is only available when editing a step that already has an ensuing step (so first create ensuing step)
    - selection the “go to next step” option in a step and subsequently delete all ensuing steps, sf changes the step to automatically reject record, if the step criteria are not met
    - selecting the “go to next step” in the first step when the record does not meet the criteria for any of the steps in the approval process, rejects the record
* Parallel approval process
  + can send approval up to 25 different users simultaneously
* Dynamic Approval Process
  + Used to route records for approval based on complex approval matrices
  + Used to route approval requests to users listed in lookup fields on the record requiring approval
  + Steps
    - create a lookup fields on the object being approved
      * uses lookup relationship
      * if it requires 3 approvers, then create 3 lookup relationships
    - Create a custom object as an approval matrix
    - Populate the approval matrix
    - Create Apex code to fill in the lookup fields from the approval matrix
    - Create or update an approval process to utilize the new lookup fields
* Automated processes
  + occur in the following order: Validation Rules->Assignment rules->Auto-Response rules->Workflow rules->Escalation rules->Parent roll-up summary field formula recalculates->Criteria-based sharing evaluation
  + Sequence in which automated processes occur may affect the outcome of a record

## 6. Increasing Data Quality Using Visual Workflow

* Used to automate business processes; allow combining one or more forms, business rules, backend calls to APIs without any code.
* Build flows to navigate a user through a series of screens
* Create branches in flows based on user input and execute logic in a flow
* Flows can query and update sf data, only users with “Manage force.com flow” permission can create and edit flows.
* Key components to create a flow:
  + Elements:
    - Building blocks of flows
    - Represent actions
    - Connect with other elements to create series of screens
  + Resources:
    - Store and manipulate data
* Tasks required to create a flow
  + Adding resources
  + Adding and defining elements
  + Setting the start element
  + Connecting the elements

## 7. Auditing Processes

* Setup Audit Trail
  + tracks changes made to the setup of an org
  + lists the date of the change, the name of the user who made the change and a description of the change
  + displays 20 most recent changes
  + tracks changes for 180 days
  + can choose upto 20 fields per object for tracking changes
* Field History Tracking
  + allows to track the history related lists for cases, contacts, leads, opportunities, solutions, accounts, contracts, and custom objects
  + modification to any standard or custom field, whose history is set to be tracked, results in a new entry in the History related list
  + for most field types, both the old and new values are captured in the History related list; however those values are not tracked for long text area and multi-select picklist type fields
  + tracks changes for upto 20 fields
* 3 tools
  + debug logs, setup audit trail, field history

**7. Data Management Overview**

* Id – first 3 chars identify the object – account ,contact, custom obj, …
* can access id through – URL, Report, Web Services API, Formulas
* Format of record ids:
  + 15 digit case sensitive
  + a column in report is displayed as 15 digit
* 18 digit case insensitive
  + web services api always return 18 digit
  + the api always returns 18 digit
  + the report framework doesn’t expose IDs for all objects
* System fields
  + Created Date, Created By, Last Modified Date, Last Modified By – these fields can be set only during the initial setup
  + only accessible through API and backward compatible with all SOAP based APIs
  + available to all custom objects, but restricted to account, opportunity, contact, lead, case, task and event standard objects
  + for updates, api will accept either the 15 digit or 18 digit

**8. Data Management Tools**

* Tools to migrate data
  + Import wizards
    - Are easy to use; do not require any programming or developer skills
    - can load up to 50,000 records – accounts, contacts, leads, solutions or custom objects
    - can import only one record type at a time
  + Web services API
    - can load data to any object supported by API
    - can load more than 50,000 records
    - can load data to any object supported by API
    - can load more than 50,000 records
    - can schedule regular data loads
    - export data for backup
    - delete multiple supported objects at the same time
    - tools include: Data Loader, Partner Tools, Custom-built Tools, Open Source Tools
* Data Loader: is a Salesforce product
  + can be run from command line or be run in batch mode
  + support custom relationships for upsert
  + supports importing from and exporting data to a CSV file
  + supports loading from and exporting data to a database through JDBC
  + available for download by System Administrator in Unlimited Edition, Enterprise Edition & Developer Edition, also available as open source but no support
* Export – uses SOQL to export records from SF to CSV
* Insert – inserts new records
* Update – updates existing records and matches records based on the Salesforce id
* Upsert – insert + update, matches based on either Salesforce id or external id
* delete – deletes records from, matches based on Salesforce id
* Bulk API
  + Is used to load high-volume data
  + Can load over 50,000 records
  + Is optimized to perform insert, update, upsert, or delete operation on large number of records
  + Improves throughput when loading large data sets due to parallel processing

**9. Extend Analytics**

* Custom Report Types
  + Are predetermined combinations of related objs. & their fields that are used as starting points when building new custom reports.
  + Determines filters avail. on report; also determines default columns, objs & fields avail in report
  + Filters are based on primary obj in report type
  + Allow to select combinations of up to 4 related obj.
  + Use it to define “with or without” relationships which are outer joins.
  + Modifying Report Builder:
    - Create, remove, rename, reorder sections
    - Remove, reorder, rename fields
    - Add fields related via lookup
    - Select which fields appear as columns by default
  + Creating a CRT:
    - Define CRT
      * Select primary obj.
      * Enter label, name, desc
      * Select cat.
      * Select deployment status
    - Add related obj & define relationships
      * Choose up to 3 related objs
      * Choose “with” or “with or without” relationships
    - Modify fields avai. for reports (optional)
      * Add, remove, move fields
      * Change field labels
      * Choose “selected by default” field
      * Modify sections
      * Add fields related via lookup
* Standard Report Types
  + Sf auto generate standard report types for new obj & its related obj.
  + If add custom fields to a standard or custom obj, custom fields are auto added to standard report types
  + Standard report types show “with” relationship which are inner joins.
* Exception Reports with Cross Filters
  + Exception report uses a “without” relationship to show where data doesn’t exist
  + Cross filters enable filtering of parent records in a report by their related child records, using WITH or WITHOUT conditions
  + Can have up to 3 cross filters per report
  + Subfilters provide an option to extend a cross filter by including only child records that meet certain criteria.
  + Each cross filter can have up to 5 subfilters.
* Categorizing report data with bucket fields
  + Bucketing enables segmentation of a report on the fly, by defining a set of categories or “buckets”, to sort, group, or filter records.
  + Create a bucket field to categorize records based on their values in picklist, number, text fields
* Summarizing Multiple Groupings on report: using PREVGROUPVAL
  + Is used in custom summary formulas to return value of a summary field from a previous grouping
  + Useful when report is grouped by a date field into periods, as summary values can then be compared to previous values to analyze trends
  + Has 3 arguments:
    - Summary field denotes name of grouped row or column.
    - Grouping level denotes summary level
    - Increment denotes number of previous rows or columns (default is one if not specified)
* Summarizing a grouping’s share on a report: using PARENTGROUPVAL
  + Allow to return value of a summary field from a specified parent grouping or grand total.
  + Has 2 arguments:
    - Summary fields
    - Grouping level
* Dynamic Dashboard
  + Displays data of user viewing it, rather than data of a specified running user.
  + Authorized users can change running user directly from dashboard view page.
  + Dashboard filter allows viewers to change data visible on dashboard by selecting a filter value from a drop-down list.
  + Can add up to 3 filters to a dashboard, each with up to 10 filter values
    - Jhfdj

**1. Introduction to Reports**

* Reports can include both standard & custom objects
* two users with different access levels see different data with the same report
* objects & fields are available immediately after they get created
* reports run in real-time data
* reports are saved in folder, report security is determined by folder where report is stored.
* System administrator and users with ‘Manage public reports’ permissions can manage the folders
* Custom Report
  + 3 panes – fields, fitlers & preview
  + 4 types of reports:
    - tabular – simple listing of data
    - summary – sorting and sub totalling
    - matrix – summarizes data in grid (to compare related totals)
    - joined – can contain data from multiple standard or custom report types
  + can customize by adding more columns, but cannot deselect action & name columns
  + the way the type of report to be created can be identified by:
    - Select Data
    - Select Report Type
    - Select Summaries
    - Select Groupings
    - Select Filters
  + scheduled reports run at user’s timezone who scheduled it
  + to run a report, the report must be in a public folder and the user has to have access to it
  + Hide Details – will show only summary rows, Show Details will show all the data, report shows only upto 2000 reports, beyond that the user can export to excel
  + Printable view – exports data to excel with the report formatting
  + Export Details – exports raw data to excel (no report format will be retained)

**2. Introduction to Dashboards**

* Dashboard – contains components based on report or chart, can use VF pages to present data, shows data as of the last time it refreshed, can be refreshed manually
* dashboard can have upto 20 components
* can choose either 2 or 3 columns
* dashboard components
  + chart – good for comparisons, is based on report data and the field which you summarize the report is the field that you see on the dashboard
  + table – shows the top/bottom n records, can be used to display totals also, can include upto 4 columns
  + guage – shows progress towards goals, display percentage or total
  + metric – shows single number (grand total)
  + visual force page – pulls data from another source
* can follow individual components through dashboard alerts and snapshots
* can post component’s snapshot to dashboard feed to chatter or dashboard feed
* chart types
  + vertical column – summary report with single grouping. Variations: vertical col: grouped, stacked, stacked 200%, good for showing dates
  + horizontal bar – good for summary with single grouping. horizontal grouped, stacked, stacked 200%
  + line – specially used to show data over time
  + donut chart – good for multiple groupings with total amount
  + funnel – good for datasets for multiple groupings in ordered sets and want to show the proportions among them
  + pie – multiple groupings and want to show proportion of single value for each groupings against total
  + scatter – illustrate degree of correlation between two axis
  + combination charts – that includes different sets of data
* dashboard security
  + dashboard folder – controls who sees the folder
  + running user – controls what data is displayed on the dashboard
* clicking a dashboard component takes users to underlying source report, filtered source report or another url that the dashboard creator specifies
* can drill down to single record, but user will see that the security model will allow to see
* report access is determined by folders
* it is possible to see a dashboard component, but not the underlying report
* Dashboard filters
  + let users chose which data to chose
  + each filter based on single field, can specify upto 10 filter options
  + enables users to view different subsets of data on the same dashboard
  + each dashboard can have upto 3 filters
  + filters can be created on date, datetime, currency, pickup, lookup and text fields
  + filters cannot be added with visual force components, scontrol conponents
  + cannot use filters for bucket fields, filter components cannot be followed on chatter
  + scheduling or emailing a filtered dashboard will return unfiltered data
* Dashboard display info as of date
* can be refreshed daily, weekly, monthly

**3. Custom Report Types**

* Standard Report Types
  + created, when an object is created, allows to report relationships between objects are created (similar to inner joins)
  + the report types cannot be modified
* Custom Report Types (CRT)
* unique templates for creating reports, created by administrators or users with ‘Manage Custom Report Types’
* can include standard or custom objects, allows users to select the objects and fields that should be related for reporting purposes
* Enable the creation of “with or without” reports
* Three steps to create a report based on CRT:
  + select the primary object
  + select related records from other objects (optional)
  + can chose upto 4 objects, chose “with or without” for each related object (similar to outer join)
  + Add fields related via lookup (optional)
* traverse multiple objects
* use lookups to join other objects
* can go upto 4 levels deep, supports many to many relationships
* primary object cannot be changed after the CRT is created
* there is a limit on # of CRTs  (depends on SF edition)
* counting rows
  + standard reports show all rows from both the objects (positions & job applications)
  + what if you want to see only unique rows so that within a report of both job applications & positions, you could see how many job apps you have or how many positions you have.
  + in this case, need to create a report that uses custom field to  count the # of records
  + two step process
    - create a new formula field
    - run a report that utilizes this new field
* to create custom summary formula: 3 steps
  + create formula fields to count the number of records for each object
  + create custom summary formula to calculate the ratio between the number of records for each object
  + optional: create a dashboard component that will show the ratio between the two numbers
* Bucketing – helps qualifying the data (for e.g. Not qualified, Qualified, Highly Qualified – for candidates)
  + create a new formula field to group the information
  + create or update a report that utilizes the new field
  + create or update a dashboard component using the new or updated report

**4. Analytic Snapshots**

* provides trending, can be scheduled
* schedule the report and capture the data in custom object and report on the data in the custom object to view historical data and analyze trends
* can be created in 3 steps:
  + create source report (tab/summary)
  + create the target object (custom). Fields in target object should have same type as in the source object that included in the source report
  + setup the analytic snapshot
  + select the source report, select the target object, map the fields on the report to the fields on the custom object and schedule the frequency for taking the snapshot

**5. Going Beyond Salesforce Reports**

* exception reports are available in a few places, but not everywhere
* no reporting on multiple related lists (using crt, you can include custom objects that are hierarchically related
* there is only one layout or UI for the reports
* Salesforce reports don’t provide trending capabilities beyond analytic snapshots
* Salesforce reports provide limited analysis of what changed bet two dates
* can’t include data in sf report from another source
* RaaS -that uses Salesforce as a source – then reports on the data by running multiple queries
* BIaaS – copies all data to local repository, then runs queries off of the copy of the data  (data repository is provided as a service)
* Data Warehousing – similary to BIaaS, but you own the servers

**III. Data Management**

**2. Basics of Upsert & External ID**

* Upsert – insert + update
* External Id
  + user defined cross reference field
  + can be created for any custom field of type text, number or email
  + helps improve report & API SOQL performance
  + each object can have upto 3 external ids
* if the external id is matched multiple times, an error is reported

**4. Managing Data**

* command line
  + can set the config directory
  + data loader runs whatever operation, file or map that is specified in the config file
  + runs the current directory if no config diretory is specified
  + default config file location: c:\program files\salesforce.com\data loader\version\samples\conf
  + if you use process-conf.xml, setting process.name to the name of a process specifies a process to run. Otherwise, the config.properties file is used for parameter settings
  + supports extract, insert, update, upsert, delete
  + offers encryption utility: Run\bin\encrypt.bat
    - Generate a key: key text is generated onscreen from the text provided
    - Encrypt text: (key file can be provided optionally)
    - Verify encrypted text
  + mass transfer tool to upsert mass data – can be used to transfer multiple accounts, leads from one user to another
  + need ‘Transfer record’ and ‘Edit’ permissions
  + to transfer a record that a user doesn’t own, the user needs to have the required user permissions and read sharing acces on the record

# VI. Visual Force Pages

## 1. Introduction to Visualforce Pages

* 2 types of UI
  + Page Builder
    - UI generated automatically
    - limited/no control of UI behavior
    - limited control over look and feel, but all UIs are consistent
  + Visualforce
    - UI generated by developer/technologist
    - full control of UI behavior
    - full ‘pixel level’ control over UI
* Visualforce & Apex
  + closely tied
  + PE/GE edition limitations prevent from authoring own apex (app exchange apps is okay)
* Visualforce inline editor
  + auto-completion
  + full syntax highlighting
  + online doc
  + can be edited through force.com id
* Developers can include
  + VF tags, Force.com expressions, HTML, Javascript, Flash
  + VF pages are limited to 15 MB
* view State
  + maintains state across multiple pages or server calls
  + view state inspector
    - shows components contributing to view state
    - must be enabled on user profile
    - is displayed only when using <apex:form>
  + view state limit is 135 kb
* vf pages
  + understand Salesforce metadata
  + display the same performance as statndard sf pages
  + are automatically upgraded to the next sf release
  + vf conforms to mvc development patter
* MVC
  + Model – standard or custom object
  + View – pages that are presented to the end user
  + Controller – that determines the logic what happens initates an action such as clicking on a tab, etc.
* 3 key elements
  + Visualforce pages
    - design definition of an app’s user interface
    - implemented using standard web technologies like HTML & javascript
    - can dynamically detect device and associate them with specific design definitions
  + Visualforce components
    - can be standard or custom UI components
    - over 65 standard sf ui elements available at G
    - referenced via a tag library model
  + Visualforce controllers
    - ability to reuse any standard Salesforce UI behavior like new, edit, save, etc (standard controller) and have access to Salesforce data
    - ability to define new UI behaviors and navigation using apex (custom controller)
* Visualforce Components
  + pre-built UI constructs which reference standard elements in the Salesforce UI
  + referenced in a VF page using an XML tag
  + dynamic visualforce components
    - are designed in apex
    - allows to create pages that render based on variety of states, such as user’s:
      * permissions, behavior, org preferences, data attributes
    - are not intended to be the primary way to create new vf pages
* Controllers
  + contain the logic and data references a page uses
  + can be used to maintain state across page interactions
  + are refernced or used by pages, through components that call data or actions
  + each page can reference or use standard controller, custom controller or custom controller extensions
  + each vf page references one main controller
* types of visualforce components
  + standard controllers
    - are available for all API entities/objects as well as custom objects
    - provide access to standard sf data and behavior
    - are referenced by using <apex:page standardController=”Contact”>
  + custom controllers
    - are coded to create custom behaviors or non standard data sets
    - can be used to create wizards or leverage callouts
    - are invoked by using <apex:page controller=”MyController”>
  + cusom extensions
    - add custom behavior or additional data to standard controllers
    - are invoked by using <apex:page standardController=”Contact” extensions=”MyClass, MyOtherClass”>
* Expressions and Data Binding
  + uses expression syntax to bind components to sf data and actions in the page’s controllers
  + expressions are linked back to controller data and actions not just to sf in general
  + all content in {!…} evaluated as an expression
  + User.FirstName} shows the current user’s first name in a page
  + data context is provided to controllers by the ID parameter, just as in standard pages.
* Versioning
  + backward compatible
  + each vf page is saved with version settigns for specified version of api as well specified version of visualforce
* Visualforce namespace
  + standard tags begin with the word apex
  + custom tags begin with the letter c
  + user can register custom namespaces to be displayed with custom tags instead of the letter c
* Incorporating VF pages in Salesforce UI by
  + creating links to reference the unique page URL
  + overriding standard buttons to route to the new page
  + overriding tab overview pages to use the new page
  + creating custom tabs for the new page
  + embedding pages into page layouts
  + adding pages to a dashboard
  + using pages as custom help for a custom object

## 2. Visual Force Tags

* Tags
  + consists of library of tags
  + can incldue text, html, javascript tags
  + can’t use javascript commenting
* Tag Rules
  + are hierarchical
  + must be closed in the reversed order they were opened
  + like xml, vf must be well-formed
  + VF and JSP
  + similar to JSP
  + typically begins with <apex>
  + all pages must be enclosed by a set of <apex:page> tags
  + components may contain attributes with values to help further define them
  + vf components are resolved into other code at runtime
* Tag Bindings
  + Bindings related visual force components with the page controller or other page components
  + 3 types of bindings
    - data bindings – use expression systan to pull the data from dataset made available by the page controller
    - action bindings – uses expression syntax to call action methods for functions coded in page controller
    - component bindings – compnent attribute values to reference other components
* Tag Data Binding
  + binding goes both ways – read and updated
* Expression syntax
  + dynamic object data can be inserted using {!objectname.fieldname} syntax
  + global data can be inserted with the added $ syntax, such as
    - User.fieldName}, {!$Page.otherVisualforcePage}, {!$Component.otherVisualforceComponent}
  + local variables can be created to stand in for these expressions as they can become long and unwieldy using the <apex:variable> tag.
* Action Binding
  + set of actions available through the controller
  + can be called using expression syntax
  + they can be
    - standard actions
    - custom actions
* Component Ids
  + all vf tags have an optional id attribute
  + this id is used as the DOM id when the page is rendered
  + the tag can be referenced by the id by other tags, javascript, or other web enabled languages
  + the ids should be unique within each page
  + the hierarchy of ids should be specified if the ids are not unique
  + when components (such as tables and lists) support iteration over record collections, the system appends \_index to the id for each row, starting with zero.

## 3. Basic Page Components

* Layout Components
  + provides a structure to the page
  + provide templates or frames to insert content
  + do not bind directly to sf data
  + are focused on areas where data-bound components can be placed
  + tags
    - apex:page /> – represents a single vf page
    - apex:variable /> – provides a local variable that can be used to replace an expression to reduce long and repetitive text
* Static  Resource Components
  + a type of sf storage
  + designed for use with vf
  + items required by the vf pages (such as javascript, css, images, etc…)
  + referenced using $Resource global variable
  + recommended method over uploading these files to document tab
  + are uploaded via Your Name|Setup|Develop|Static Resources
  + can be contained in an archive (zip)
  + limited to 5 MB per file and a 250 mb overall
  + use action attr to redirect
  + Components
    - <apex:stylesheet> – to add additional css file
      * are located in /sCSS/directory
      * dStandard.css – most styles for standard objects/tabs
      * allCustom.css – styles for custom objects/tabs
    - <apex:pageBlock>
      * helps build out pages and uses sf stylesheet by default
      * creates an area of a page that is similar to detail page and doesn’t contain the default content
    - <pageBlockButtons> – set of buttons that are styled like standard sf buttons (buttons still need to be created manually)
    - <pageBlockSection> – must be used within a pageBlock component. This tag creates a section with one or more columns
    - <pageBlockSectionItem>
      * used within pageBlockSection component
      * allows to modify data presentation, display the data using a different widget, or present items not based directly on SF object fields
      * if we need to bundle more than one item in the cell, then use outputpanel
    - <apex:sectionHeader> – creates the standard colored header bar displayed under the tabs in the SF UI
    - <apex:toolbar>
    - <apex:toolbarGroup>
    - <apex:tabPanel>
    - <apex:tab>
    - <apex:panelBar>
    - <apex:panelBarItem>
    - <apex:panelGrid>
    - <apex:panelGroup>
* Coarse Metadata Components
  + provide a large amount of generated code to create familiary Salesforce structures
  + do not allow for much customization to the generated areas
  + Components
    - <apex:detail />
    - <apex:relatedList />
    - <apex:listViews />
    - <apex:enhancedList />
    - <apex:repeat />
  + Chatter tags
    - enable to add chatter into vf paes
    - incorporate chatter into vf pages using
      * the showChatter attribute of <apex:detail> tag
      * <chatter:feed> to include chatter feed on a record
      * <chatter:followers> to include chatter followers on a record
      * <chatter:feedWithFollowers> to include chatter feed, followers and show/hide chatter button
      * <chatter:follow> to add a button that enables you to follow records
  + Message components
    - <apex:pageMessages> – use the standard sf style
    - <apex:messages> – unformatted but can apply custom style
    - <message> and <pagemessage> – specific to one component
    - messages always shows up in system log.

## 4. Form and Output Components

* allow entering info into your pages, & uploading files
* form components
  + <apex:form>
    - enables a section of a vf page to allow users to enter data and subit it with commandButton or commandLink
  + <apex:inputField>
    - corresponds to a SF object field that respects the attributes of that field and uses associated sf UI widget
  + <apex:inputWidget>
    - set of widgets for data that doesnt correspond to a SF object field to be used with pageBlockSectionItem tags
      * <apex:inputCheckBox>, <apex:inputHidden>, <apex:inputSecret>, <apex:inputText>, <apex:inputTextarea>
    - limitations
      * no protection from malicious javascript
      * no escaping/unescaping the data correctly when displayed on a regular page layout
      * no built-in handling of the truncated display of long text fields
      * no special search indexing to ignore tags and focus on the plain text
      * no special handling of the field when used in filters, workflow rules, etc.
  + <apex:selectWidget>
    - series of additional tags to support the display of UI widgets in organized tables
      * <apex:selectCheckboxes>
      * <apex:selectList>
      * <apex:selectRadio>
  + <apex:inputFile>
    - allows users to upload files and turn them into attachments on records, documents  or apex blobs
  + <apex:commandButton> & <apex:commandLink>
    - used within a form tag.
* output components
  + display info without allowing the user to change any data
  + have parallel form components
  + components
    - <apex:outputLabel> – creates a label for input or output widgets that do not automatically come with a label
    - <apex:outputField> – creates a read-only display of a label and value for a SF field, automatically formatted according to the field type
    - <apex:outputLink> – creates a link to URL
    - <apex:param> – used as achaild tag that provides a name/value pair parameter for its parent compoentn. It can be used with
      * outputLink: defines http query string parameters
      * outputText: defines text insertion parameters
      * actionFunction: defines javascript function parameters
    - <apex:outputPanel> – tag defines a set of content that is grouped together (often for ajax)
      * layout attribute: block, inline, none
        + block: Generates an HTML div tag (adds a paragraph)
        + inline: Generates an HTML span tag (default:doesn’t do any formatting)
    - <apex:outputText> – displays text which can be formatted using a stylesheet
    - <apex:pageBlockTable> – creates a table by iterating over a set of data using the SF stylesheet. good if data comes from sf object. used within pageBlock or pageBlockSection
    - <apex:dataList> – creates a list (a one-column table) by iterating over a set of data
    - [Note: dataList and dataTable are very similar and generally used when you don't want the standard sf table style. DataLists are just one-column tables.
    - <apex:dataTable> – creates an HTML table which iterates over a set of data
    - <apex:column> – used within either pageBlockTable or dataTable set of tags. it creates the columns for a table
    - <apex:flash> used to embed flash widgets into a vf page
    - <apex:facet> used with a variety of other component tags to provide or override headers, footers, and captions to other items

## 5. Visual Force Components for Modularity

* Custom components can stand alone or be accompanied by a custom controller (can be shared in appexchange)
* <apex:component> – used to create our own custom reusable components
  + access then using <c:componentname>
* <apex:attribute> – use it within component tag to define the custom attributes, can define the name, data type, and other aspects of the custom attribute
* <apex:componentBody> – used within a component tag to pull the body of the component’s implementation into the component definition, often used for custom iteration component
* Page Inclusions (mashups)
  + <iframe> to include another page as URL
  + <incldue> – to include another vf page
* Template Tags
  + series of tags that are used to create vf template pages and define reusable components for baseline pages
  + tags
    - <apex:define>
    - <apex:insert>
    - <apex:compositions>
* <messaging:emailTemplate>
  + facilitate communication outside of the application
  + used to create vf email templates
  + must be wrapped inside a single emailTemplate component
  + have advantages over  traditional email templates
  + can be edited using Email Templates (under admin)
* Messaging tags
  + <messaging:emailHeader>
  + <messaging:htmlEmailBody>
  + <messaging:plainTextEmailBody>
  + <messaging:attachment>
* With email templates, you can
  + repeat tag to iterate through all of the related records
  + generate pages inside of the template
  + specify a custom email header
  + create attachment using plain text, HTML or VF
* Visualforce Performance Troubleshooting
  + reduce view state size using only one <apex:form> tag on a page
  + cache frequently accessed resources
  + reduce page size < 15 mb
  + increase the time interval for calling apex from visualforce page  
    remove unnecessary fields to reduce the amount of data returned

## 6. Javascript in Visualforce

* Action Binding and Javascript
  + currently only actions that are shared across al objects are exposed through standard controllers
  + but further standard sf actions are available by using javascript and the expression syntax with the !URLFOR and $Action keywords
* Ajax tags
  + 5 tags
    - actionStatus – used to display start and stop statuses of ajax requests
    - actionSupport – used to call 2nd component when an event happens to the 1st component
    - actionPoller – similar to actionSupport, but the event is based on timer instead of a user action
    - actionFunction – provides support of invoking a controller action from javascript code using an Ajax request by defining new javascript function
    - actionRegion – used to demarcate which parts of the page the server should reprocess
* use rerender attribute to do partial updates
* simple to implement partial page update is
  + isolation the portion of the page by surrounding it with <apex:outputpanel> tags. be sure to give id attribute
  + create the command button or link that will trigger the partial refresh. add the rerender attribute and assign it the value of the id of the outputPanel created earlier
* if event happening to same component that should action, use the built-in javascript event attributes
* if event happening to a different component that will take the action, use the actionSupport tag to handle the event
* With ajax toolkit
  + create an apex class and expose it as a web service
  + call the web service from a visual force page
    - optionally can attach a page to a button, make it inline, etc.

## 7. Reports and Dashboards

* Use reports to analyze and display data in different ways

## 8. Further Topics in Visualforce

* force.com sites allow to build public unauthenticated sites that can access data from sf apps
* 4 main use cases
  + build and run new web applications
    - consumer reviews, hotel conceirge services, event registration sites
  + transform business apps into websites
    - recruiting portal
  + extend your salesforce crm apps
    - interactive web to lead forms
    - campaign landing pages
  + run your corporate web site on salesforce service
    - public websites, intranets
* Salesforce Mobile
  + licensed client app that can be run on blackberry, iPhone, or windows mobile device
  + provides mobile access to data, email, tasks and calendar
  + includes features such as permissions, page layouts, related lists, dashboards, reports and list views
  + allows administrator to mobilize a limited set of standard objects and all custom objects
  + lite edition is free
* guidelines to develop pages for mobile
  + evaluate if app interface needs to be redesigned for the use on mobile devices
  + keep the real estate open by not displaying the header or sidebar
  + avoid using lookup fields. For the best user experience, use apex to validate data entry
  + create reusable styles in a separate page and use the include component to add these styles
  + use a third party libary such as iUI that provides iPhone like interface
  + refrain from createing styles as a static resource
* iPhone
  + set page width to 980 pixels
* Blackberry
  + doesn’t support inline events
  + doesn’t have built-in navigation
  + viewstate for forms is too large for Blackberry
  + use standard html forms in mobile page instead of using form component
* 3 methods to develop for multiple platforms
  + Separation and redirection
    - build pages separately and point the mobile tab to the bb page
    - top of the page, include the js to redirect the page, if the target is not a bb device
  + Lowest Common Denomiator
    - create pages that include minimal or javascript
    - use these pages on any supported device
  + Conditional Code
    - create pages that evaluate which device being used
    - offer appropriate markup for each device
* Mobile Javascript Librar
  + some functions of mobile devices not applicable to desktop clients
    - developers can use js functions in vf pages for javascript enabled devices
      * mobileforce.device.sync()
      * mobileforce.device.close()
      * mobileforce.device.syncclose()
      * mobileforce.device.getLocation()
    - html links can be used to sync/close
* Mobilizing Visualforce Pages
  + Create new mobile ready visualforce tab
  + add the vf tab to mobile configuration
  + test the page using a mobile client simulator
* Chatter Data Model
  + FeedItem is the fundamental entity for the chatter data model
  + Feed tracking can be enabled for upto 20 fields per object