**CONTROL PAR MODEL KA OBJECT**

**ANS-- marksheetmodelint model = modelfactory.getinstance().getmarksheetmodel();**

**1 . What is new in this project**

We have use hibernate , bootstrap , and add jasper report functionality in our project.

**2. Lazy fetching**

True – it means it will brings only recommended data or data on demand.

False – it will brings all the data even FK table data also.

**3. Eager fetching**

Fetch= ”join” - it means single join query will brings all the data.

**4. batch fetching**

It is by-default “true”. it will brings or fetch the data with the help of PK & FK.

**5. how to apply navbar**.

<nav class=”navbar navbar-expand-lg”></nav>

**6. does navbar follows the grid system or not?.**

Navbar doesn’t follows the grid system because it has the fix number of girds (columns).

**7. what is hibernate.**

Hibernate is a pure java objects oriented mapping and persistent framework that allows us to map plain old java object to relational database using configuration files.

**8. jasper report**.

It is an open source java reporting engine . it is a java class library and it meant for those developer who needs to add the reporting capabilities in their application.

**9. lifecycle of jasper report .**

* Designing the report
* Compiling the report
* Executing the report ( filling the data into report)
* Exporting the reporting to desired format.

**10. how did you create jasper report**

Using TIBCO tool.

**11. classes for jasper report.**

* Jasper report
* Jasper fill manager
* Jasper compile manager
* Jasper print manager
* Jasper export manager

**12. what is bootstrap**

Bootstrap is the most popular HTML , CSS , and java script framework that help us to develop responsive web pages and mobile friendly websites.

**13. what is model factory ?**

It is the factory of all model classes. It follows the factory design patterns & singleton design pattern.

If there are multiple implementations then it provide user to access that which implementation class to instanciate.

**14. how many DCP we have used in our application**

1 DCP . and it is based on how many database we are using.

**15 . Design patterns .**

* Factory design pattern – the class has ability to create an object of other class.

USE - In session object creating. Eg,

* Front Controller design pattern – it prevent any user to access application without login

Use – FrontCtl

* Bridge design pattern - single interface , multiple implementation

USE – model factory

* Builder design pattern – create a complex object using simple object using step by step Approach .

USE – EmailBuilder

* Singleton design pattern – the class only one instanace in lifetime.

`1. Make class final

2. declair a static variable

3. define a private constructor

4. define static getInstance method.

**16. Session Factory**

It is an interface . it is created by config setting in hibernate.cfg.xml file .

It is a DCP that creates only one object in their lifetime for a database.

**17. MYSQL dialect.**

It converts HQL query to our native query language (mysql) .

**18. Root tag of hibernate.cfg.xml file**

<hibernate – configuration>

**19. use of <mapping resource> tag**

It contains all the hbm.xml files mapping path.

**20 . how to show statement on console .**

Show\_sql = “true”

**21. Inheritance in hibernate .**

* Table per hirarchary - single table is require to map whole hiararchry & discriminator

Coloumn is added to identify the class.

* Table per concrete class – it means the created table has no relation with other table.
* Table per sub class – A class is created when it has parent and child relationship

Such as PK % FK.

**22. DOCTYPE –**

It is a document type declaration that contain information about the particular XML file.

**23. how to create Session Factory object .**

SessionFactory sf = new configuration().configure().buildSessionFactory();

**24. how to create session object .**

Session session = sf.openSession(); - when we want to manually handle the transations.

=sf.getCurrentSession(); - when we want to automatically handle the transation.

Disadvantage – it can be store NULL value in DB.

**25. Object state**

* Transient – only object created but not associated with session & PK.
* Persistence – after opening session and just save data into DB.
* Detached – when session in closed & just state of object.

**26. how to convert form detached to persist -**

After closing the session the object will be in detached state . so we must open a new session and call session.update(); .

**27. evict();**

Removes only one particular object from the session.

**28. close();**

Use to close the current session.

**29. clear();**

Clear all the data from the session.

**30. how does transation handling.**

Transation ts = Session.beginTransation();

ts.commit();

Ts.rollback();

**31. transation –**

It is the logical unit of work for the data transation & the most RDBMS support.

**32. Cascading –**

It remove the child object when parent object is removed.

**33. key interface of hibernate** .

* Configuration
* Session factory
* Session
* Transation
* Query and criteria

**34. hibernate.cgf.xml –**

It is a configuration file that contain all the hbm.xml file mapping and contain property that is used for estabilish the connection with the database.

**35. how did we use hibernate .**

Firstly we have add hibernate dependency in POM.xml file & create a hibernate.cfg.xml file for get a physical connection with DB.

**36. Scope –**

* Application
* Page
* Session
* Request

**37. Criteria –**

It is an interface for retrieving entities by composing criteria object.

Used for search operation.

* Criteria – attached with session at compile time
* Detached criteria – attached with session at run time .

Create query & initial creation without session

We must convert into executable query

* getExecutableCriteria();

**38. Criteria Vs HQL –**

Criteria – only for selective / Situable for dynamic query / Pagination support

HQL – both selective & non-selective / situable for static query / doesn’t support pagination

**39. how to apply pagination .**

Set on model – criteria.serFirstResult( (pageno-1)\*pageSize);

Criteria.setMaxResult(pageSize);

**40. who creates table into database.**

<hbm2ddl auto>true

**41. utility classes.**

DataUtility / DataValidator / EmailBuilder / EmailMessage / EmailUtility / PropertyReader

ServletUtility / HTMLUtility / JDBCDataSource / HibDataSource

**42. DEFFERENCES**

GET LOAD

* fetch data from DB on every request fetch data from session
* if record not found . it return “NULL” if record not found it return recordNotFound
* return Real data return Proxy data

**SAVE PERSIST**

* returns serializable object reutrns void
* it means how many query save in DB.

**UPDATE MERGE**

* firstly it checks that the data exist it update without checks session & hit the DB.

in the session

**1st LEVEL CACHE 2nd LEVEL CACHE**

* by-default enable developer must be do enable
* associated with session associated with session factory.

**43.what we did for use bootstrap**

We have add bootstrap library

* bootstrap.min.js 4.4.1
* bootstrap.min.css 4.4.1
* jquery.min.js 1.12.4

**44.librabry for calender**

* jquery-1.12.4.js
* jquery-ui.js 1.12.1

**45 how to add script and css files .**

For Script <script></script>

For CSS files <link/>

**46 library for fonts and icons**

Font awesome library 5.6.3

**47. input validation**

If( dataValidator.isNull(“name”){

Req.setAttribute(“name”,propertyReader.getValue(“error.require” , “name”);

Pass = “false”}

For get servletUtility.getErrorMessage(“name”, request);

**48. advantages of bootstrap**

* it is very easy to use. Anybody having basic knowledge of html & css can use bootstrap
* it fecilitates user to develop responsive websites.
* It is compatible with most of browser like chrome,opera,firefox etc.

**49. what is ORM.**

ORM stands for object relational mapping. ORM is automated persistence application to the table in a relational database.

**50. ORM consist-**

* API for performing basic CRUD operation
* API to express queries to referring classes
* Facilitates to specify metadata

**51. what we did for dropdown**.

* .dropdown – indicates a dropdown menu
* To open the dropdown menu use a button or a link with a class of .dropdown-toggle and the data-toggle=”dropdown” attribute.
* .caret - is use for dropdown icon indication
* .dropdown-menu – class to a <ul> element to actually build the dropdown

**Example –**

<li class=”dropdown>

<a class=”dropdown-toggle” data-toggle=”dropdown” href=”#”>ROLE <span class=”caret”></span> </a>

<ul class=”dropdown-menu”>

<li> <a href=”#”> <span class=”glyphycon-plus”></span>ADD ROLE </a> </li>

<li> <a href=”#”> <span class=”glyphycon-list”></span>ROLE LIST </a> </li>

</ul> </li>

**52. what is navbar –**

A navbar is like navigation header that is placed at the top of the page.

**53. bootstrap buttons**

Btn / btn-primary / btn-success / btn-default / btn-info / btn-warning / btn-link /btn-danger

**54. what is mobile friendly website.**

Mobile first style is a part of core framework of bootstrap

<meta name=”viewport” content=”width=device-width , initial-scale=1” />

**55 how can we map whole class as immutable.**

Mutable=”false” . by-default “true”

**56. bootstrap pagination**

Use to short the webpages it becomes necessary when we have a lot of webpages.

* .pagination - use for get the pagination on our page
* .disable – customize unclickable links
* .active – indicate the current page

**57. bootstrap container**

It is used for set the contents margins dealings with the responsive behaviour of our layout.

It contains row elements and the row elements are the container of columns(grid).

* .container – responsive fixed width container
* .container-fluid – full width container.

**58. responsive websites**

Automatically adjust itself to look good on all devices.

**60. technology used in these project.**

JDK 1.8 / JDBC 3.0 / JSP 2.2 / servlet 3.1 / JAVA mail 1.4.7 / hibernate 3.0 / MYSQL 5.1.6

Jquery 1.12.4 / bootstrap 3

**61. tools that are used in these project**

* Elicpse mars
* SQLYOG
* Argo UML
* ER Win
* I report / tibco tool

**62. what is report .**

A report is well designed and summarized presentation of an information

**63. report template –**

* TITLE
* PAGE HEADER
* COLUMN HEADER
* DETAILS
* COLUMN FOOTER
* PAFE FOOTER
* SUMMARY

**64. what is show\_sql .**

It logs all the generated SQL element on console.

**65. what does Session.beginTransation();**

Begin a unit of work and returns a associated transaction object.

**66. Features of Jasper Report –**

* It has a flexible report layout.
* It can represent data either textually or graphically.
* Developer can supply data in multiple ways.
* It can supply data from multiple sources.
* It can generate watermark.
* It can generate sub Report.

**67. Hibernate Vs JDBC –**

* Hibernate – database independent

No need to take care of mapping table.

* JDBC -- database dependent

Need to create table in database.

**68. in which case Rollback Occurs.**

Lost of communication with participant in case of database transaction.

An invalid or out of range parameter.

**69. what is Proxy data.**

Ex. – proxy data returns by Session.load() method

It means hibernate will prepare some fake object with given identifier value in memory without hitting the DB.

**70. Session factory is immutable class.**

**71. Session –**

It is main runtime interface between a java application and hibernate

It is used for get the physical connection with database.

**72. Role of session interface**

* Wraps a JDBC connection
* Factory of transaction
* Holds First Level cache

**73. Dirty checking**

The automatic dirty checking feature of hibernate call update statement. If it found modification in transaction.

**74. Dynamic update & Dynamic insert –**

* Dynamic insert – it generated at runtime & insert those column which value is not NULL
* Dynamic-update- it generated at runtime & update those column which value has changed.

**75. for show the SQL statement on Console**

<property name=”show\_sql”> true </property> in cgf.xml file

**76. Named SQL query –**

Define in mapping file (hbm.xml)

<sql-query name=”empDetails” >

< return alias=”emp” class=”com.text.employee” />

SELECT emp.EMP\_ID AS {emp.empID} , emp.EMP\_NAME AS {emp.empName} FROM Employee EMP WHERE emp.NAME LIKE :name

</sql-query>

\*invoked named query -----

List people = session.getNamedQuery( “empDetails” ) . setString( “TOMBREDDY” ,name) . setMaxResult(50).list();

**77. why caching –**

Application performance optimization

A cache is designed for reduce the traffic between our application to DB.

**78. How to enable second level cache –**

<property name =”cache.provider\_class”>**org.hibernate.cache.ehCacheProvider** </property>

<property name =**”hibernate.cache.use\_seccond\_level\_cache”**> **true** </property>

**79. cache usage –**

<cache usage = “**read-only / nonstrict-read-write / read-write / transactional** “>

**80. Query cache –**

* We use query cache in the case of query is executed again and again because it holds query along with resultSet.
* Second level chache should be enabled in these case
* Need to set permission – **hibernate.cache,use\_query\_chache** set true
* Query.setCacheble(true);

Disadvantage – it will return object that is stored in cache . not to hit the DB.

**81. callback interface –**

Allow the application to receive a notification when something interesting happens with an object

Such as save / load / delete

**82. fetching strategy**

It declare that how many query will run for a relation.

**83. inverse=”true”**

It shows that the class is relationship owner

And the other class is child class.

**84. cache provider classes -**

* EH cache
* OS cache
* swarn cache
* JBoss cache

**85.how can we directly access a instance variable without setter method.**

By mapping the property with **access=”field”**

in hibernate metadata. It forces hibernate to bypass the setter method.

**86. way to express join in HQL**

* An Implicit association join
* An ordinary join in FROM clause
* A fetch join in FROM clause
* A theta-style join in WHERE clause

**87. Most common method of hibernate configuration.**

* Programatic configuration
* XML configuration

**88. Role of Stored procedure**

It is used for Time-Critical Report

**89. how to call stored produre**

* Using JDBC connection
* Using Native SQL query
* Using NAMED query

**90. candidate key**

A not null key which candidate for PK .

**91. Composite key**

A primary key which consist of two or more attribute is called composite key <composite-id>

# we have never used it because we have used non-business PK.

**92. ORM level**

* Pure Relation ( stored procedure )
* Light object mapping ( JDBC )
* Medium object mapping
* Full object mapping ( composition , polymorphism , inheritance )

**93. why we need ORM tools like hibernate**

* Improved productivity & performance
* Improved maintainiblity & portability

**94. what does hibernate simplify**

* Saving and retrieving our domain object.
* Making database column and table name changes.
* Centralizing pre-save & post-retrieve logic.

**95. what is the general flow of hibernate connection**

* Create SessionFactory object from configuration object.
* Get one session from sessionFactory.
* Create HQL query.
* Execute query to get list of certain java object.

**96. how we define sequence generated PK**

< id column=”USER\_ID , name=”id” type=”java.lang.long” >

<generator class=”sequence” >

< param name=”table”> sequence\_Name < /param>

</generator> </id>

**97. casecade & Inverse option in One-many mapping**

Casecade :- enable operation to casecade to child entities

Casecade :- all / none / save – update / delete / all-delete-orphan

Inverse :- mark this collection as the inverse end of a bidirectional association

Inverse :- “true / false”

Inverse indicates which end of a relationship should be ignored.

So when persisting a parent who has collection of children

* Should you ask the parent for its list of children
* Or ask the children who the parent are.

**98. Hibernate template –**

Org.springFramework.orm.hibernate.hibernateTemplate

It is a helper class which provides methods for querying & receiving data from the database.

It also convert checked HibernateException to unchecked DataAccessException.

**99. Benefits of hibernate template**

* A spring template class simplifies interaction with session
* Common function are simplified to single method calls
* Session are automatically closed
* Exception are automatically caught & convert to runtime exception

**100. switch between database without change the code.**

Using Hibernate dialect we can switch database

<property name=”dialect”> **org.hibernate.dialect.\_\_\_\_** </property>

**101. derived property**

The property that are not mapped to a column. But calculated at runtime by evaluation of expression is called derived property.

Define by formula attribute.

**102. Component mapping in hibernate**

* A component is an object saved as value not as a reference
* A component can be saved directly without needing to declare interface or identifier property
* Require to define an empty constructor
* Shared references not supported.

**103. sorted collection Vs Order collection**

* Sorted collection – sorting a collection by utilizing the sorting feature provided by java collection framework

Use when collection is not large.

* Order collection – sorting a collection by specifying the order-by clause when retrieval.

When collection in large.

**104. what is transactional write-behind.**

Hibernate uses a sophisticate algorithm to determine an efficient ordering that avoid database foreign key constraint.

Violence but is still sufficient predictable to the user.

This is called transactional write-behind.

**105. collection types in hibernate**

Bag / set / list / array / map

**106. HIBERNATE Vs EJB 3.0**

Hibernate EJB

* Session – cache or collection of persistence – context set of entities managed

Loaded object relation to a by entity manager

Single unit of work

* XDoclet Annotation java 5.0 Annotation
* Defines HQL Defines EJB
* Support entity relation by mapping support entity relation by java 5.0 anotation

Files

* Provides a persistence manager API provides a entity manager interface
* Entity relationship are unidirectional entity relationship are bidirectional

**107. hibernate advantages**

* Database independent - it can connect with any DB like Oracle , MYSQL
* Transparent persistence – ensure the automatic connection b/w application’s object with the database table
* Provide HQL ( hibernate query language.
* Dual layer caching
* It is an open source ORM tool.

**108. Association, Aggregation and Composition.**

* Association – it means relationship between multiple object. It refers

How objects are related to each other and how they are using

Each other functionality.

Composition and aggregation are two type of association.

* Composition - Strong association . it means if an object owns another object and the another object cannot exist without owner object.

Ex – consider the case of car having engine .

Here car object contain s the engine but engine cannot exist without car.

* Aggregation – weak association . it means object can exist independently.

Ex - consider the team object and player .

Here team contain multiple players but a player can exist without a team.

**109. benefit of ORM tool**

An ORM tool simplifies the data creation , data manipulation and data access . It is a programming technique that maps the object to the data stored in the database.

It internally uses the JDBC API to interact with the database.

**110. what is generator class in hibernate ( hbm.xml )**

The **<GENERATOR>** class is the sub-element of ID . it is used to generate the unique identifier for the object of persistent class

A generator class can be implemented by **IdentifierGenerator** interface

<generator class=”assigned / increment / sequence / identity / select”>

**111. what is a Dialect**

The Dialect specifies the type of database used in hibernate.

**112. LOG4J in hibernate**

Logging enables programmer to write the log details into a file permanently.

**113. LEVEL of LOG4J**

* OFF - used to turn of logging
* WARNING – indicates a problem
* SEVERE – indicates a failure
* INFO – used for informational message
* CONFIG – used for static configuration message.

**114. how can we change name of Hibernate.cfg.xml file**

We can Rename cgf.xml file name & pass as a parameter in Configure() method.

SessionFactory sf = new configuration.**configure(\_\_\_.cfg.xml)**.buildSessionFactory();

**115. Difference between Bootstrap 3 & Bootstrap 4**

**BOOTSTRAP 3 BOOTSTRAP 4**

* Based on 4 tier grid system based on 5 tire grid system

( xs , sm , md , lg ) ( xs , sm , md , lg , xl )

* Img-responsive class img-fluid class
* Font size 14 px font size 16 px
* Btn-xs support it has only btn-sm and btn-lg
* Primary unit - px primary unit - rem
* Pager – it use .next or .previous it use .pager-next or .pager-previous

**116 how can we make singleton class**

* make class final – so child can not be create a single class
* make a default constructor private so no one other class can instanciate.
* Declare a static variable self type because static variable have one copy in lifetime
* Define a getInstance() method that will return instance of same class.

**117. what is hbm.xml file**

It is a mapping file which instruct hibernate to map define class to database.

**118. Define the elements of generator class**

< id > < generator class=”assigned” > < /generator > < /id >

* Assigned – it is the default generator strategy. if there is no <generator> element

In the case application assigns the ID.

* Increment- for each attribute value, the hibernate increment the identifier by 1.
* Sequence – it uses the sequence of the database. If there is no sequence defined

Then it creates a sequence automatically.

* Hilo -it uses high and low algorithm to generate the ID.
* Native – it uses identity, sequence or hilo depending on the database vendor.
* Identity – it is responsible for database to generate unique identifier.
* Seqhilo – it uses high and low algorithm on the specified sequence name.
* Uuid – it uses 128-bit UUID algorithm to generate the ID.
* Select – it uses the primary key return by the database vendor.
* Foreign – it uses the id of another associated object. mostly used in <one-to-one>

Association

**119. what is Dialect**

It is an interpreter. Which convert HQL query to native query, it depends on which dialect we are using.

**120. who create table in database and how to data insert into database.**

<property name=”hbm2ddl.auto”> create / update / createupdate </property >

**121. what is preload method.**

Load the data at the time of HTML page loading in dropdown.

**122. how did you apply filter in your application.**

For apply filter in our application. Firstly we will take a input field on view page.

Then populate data on controller through populate method.

And append query in search method on model.

Ex- criteria.add(restrictions.like(“name” , dto.getName()+ “%”);

**123. how did you apply calender.**

we have added library \*jquery.min.js \*jquery-1.12.4.js \*jquery-ui.js

and enable datepicker on view page. <input type=”text” id=”datepicker”/>

**124. index formula**

Int index = ( (pageNo – 1) \* pageSize ) +1 ;

**125. how to disable next button on view.**

Firstly we get the nextPageSize through the nextListSize

Int nextPageSize = DataUtility.getInt( req.getAttribute(“nextlistSize”).toString());

And check if ( nextPageSize ! = 0 ) ? “ ” : “disabled”

**126. how to get nextListSize on controller.**

List next = model.search( dto , **pageNo+1** , pageSize );

Then request.setAttribute(“nextListSize” , next.size() );

**127. how did you get list on view.**

List list = servletUtility.getList(request);

Iterator <UserDTO> it = list.iterator();

While ( it.hasNext() ) { dto = < UserDTO > it.next () ;

**128. what will perform action when we click on edit link.**

When we will click on edit link

Request goes to TimeTableCtl doGet method and id will append in url

After that inside doGet method of ctl get the id from request parameter and create model object it will run model.findByPK() method that returns a DTO object And set DTO object through servletUtility.setDto ( dto , request ); and forword on view page.

After that on TimeTableView page create and object of DTO through jsp:useBean tag and get values.

**129. how to create an dto object on view page.**

< jsp:useBean id= “dto” scope=”request” class=”in.corays.dto.TimeTableDTO” />

**130 . Define JSP tags and elements.**

**JSP** stands for java server pages. this is special form of servlet to develop web pages.

**Lifecycle –** jspInit() , \_jspService() , jspDestroy()

**Element -** Scripting element / Declarative element / Action element

**SCRIPTING ELEMENT –** generally it is used to provide the ability to execute java statement inside the JSP page.

* Scriptlet tag – it is used to write java control statement inside JSP page.

**<% java statement %>**

* Expression tag – it is used to add result of one line expression.

**< % = java expression %>**

* Declaration tag – it is used to declare instance variable or method inside of a JSP.

**< % ! declaration 1 ; declaration 2; , …… >**

* JSP comment – used to add developer comment for developed JSP code.

**<%-- JSP comment --%>**

**DIRECTIVE ELEMENT -**generally directives are used to inject additionally programming statement in the source programing is compiled.

* Page Directive – page directive is used to configure properties of current JSP page.

**Import / info / isErrorPage / session / buffer / extends**

* Include library – this tag is used to include content of a file into current JSP page.

**< % @ include file = ” Header.jsp” % >**

* Taglib directive– JSP APIs allows you to create custom JSP tags

**< % @ tagliburi = ”tldFileUri” prefix= ”string” % >**

**ACTION ELEMENT –** generally it is used to perform some actions or task in the JSP page.

* jsp:include – this tag inject content of another resources into current JSP response

**< jsp:include page = “relative URL” flush= “true” />**

* Jsp:forward – this tag is used for transfer execution control from current JSP to another JSP.

**< jsp:forward page = “ralativeURL” | < % = expression %> ” >**

* Jsp:useBean – it is used to create bean (POJO ) object on current JSP page.

**< jsp:useBean id=”bean” scope=”request” class =”com.user” />**

* Jsp:setProperty– it is used to set values of attributes of a java bean.

**< jsp:setProperty name=”s” property=”rollNo” value=”a1”/>**

* Jsp:getProperty– retrieve value of the attribute of bean

**<jsp:getProperty name=”s” property=”rollNo” />**

* Jsp:plugin – this tag is used to embed applet in the JSP page.

**< jsp:plugin type=”applet” code=”TextEditor.class” name=”editor”/>**

**131. JSP implicit object**

* **Request / response / session / application / config / page / pageContext / out / exception.**